DOCUMENT RESUME

ED 037 097	24	EM 007 900

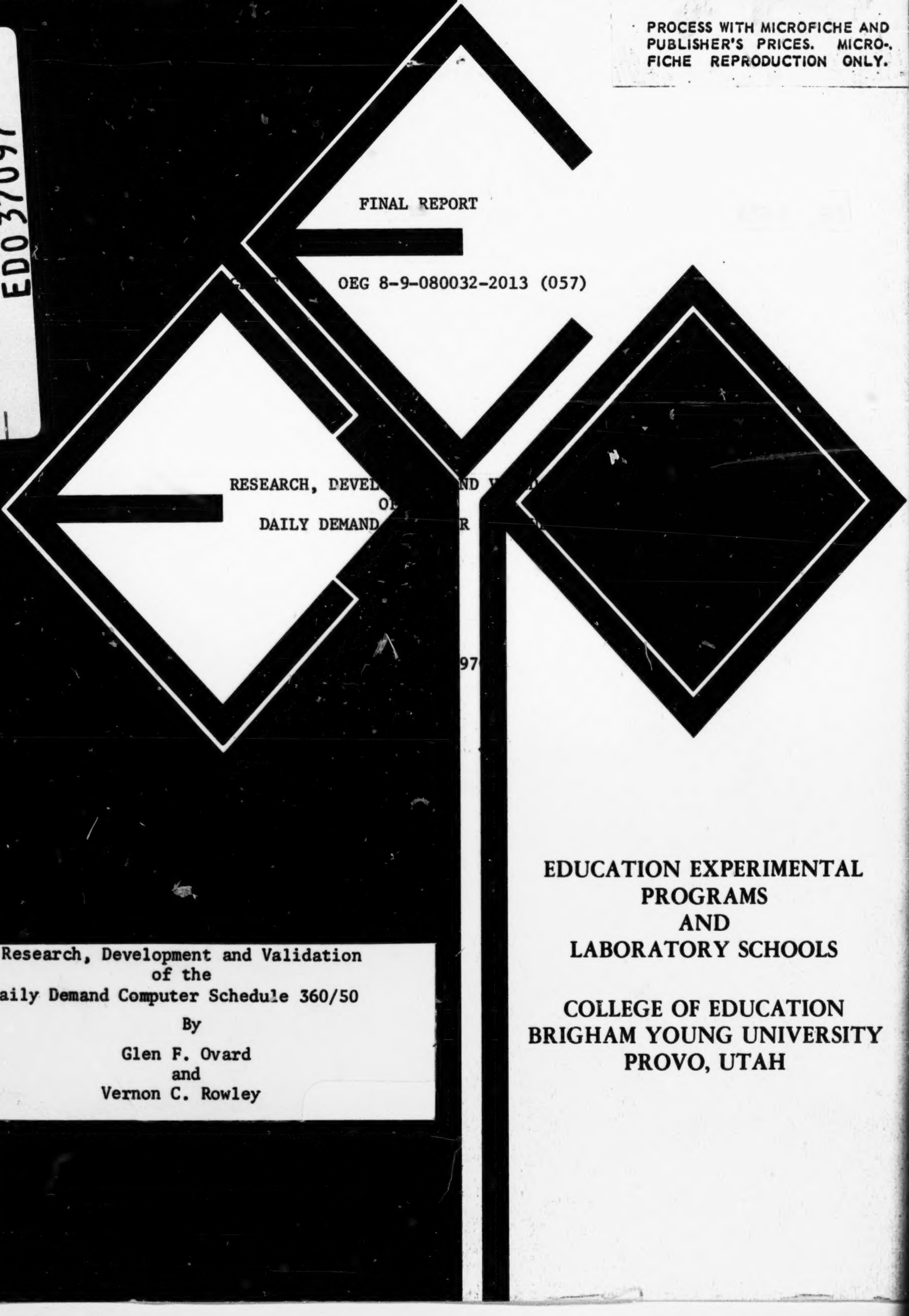
AUTHOR	Ovard, Glen F.; Rowley, Vernon C.
TITLE	Research, Development and Validation of the Daily Demand Computer Schedule 360/50. Final Report.
INSTITUTION	Brigham Young Univ., Provo, Utah. Coll. of Education.
SPONS AGENCY	Office of Education (DHEW), Washington, D.C. Bureau of Research.
BUREAU NO	BR-8-H-032
PUB DATE	Feb 70
GRANT	OEG-8-9-080032-2013(057)
NOTE	241p.
AVAILABLE FROM	Education Experimental Programs and Laboratory Schools, College of Education, Brigham Young University, Provo, Utah 84601
EDRS PRICE	EDRS Price MF-$1.00 HC Not Available from EDRS.
DESCRIPTORS	*Computer Oriented Programs, *Computer Programs, *Continuous Progress Plan, Program Improvement, Secondary Education
IDENTIFIERS	*Daily Demand Computer Schedule, DDCS

ABSTRACT

A study was designed to further the research, development and validation of the Daily Demand Computer Schedule (DDCS), a system by which students can be rescheduled daily for facilitating their individual continuous progress through the curriculum. It will allow teachers to regroup students as needed based upon that progress, and will make time a tool of the teacher and learner. Specifically, the objectives of this study were to redesign and write the DDCS 7040 computer program for a 360/50 computer, and to validate the new computer schedule with the input data previously used in 40 days of continuous operation on the DDCS 7040 program. A systems analyst and programers determined new parameters required to improve and to generalize the current program to schools of varying size and circumstance, wrote a 360 computer program, and validated the new program with input data previously used during 40 days of consecutive operation. The new program was tested and evaluated in a pilot school. Findings showed that the 360/50 DDCS system was successful, although the program requires additional testing in an operational school to attain optimum efficiency. [Not available in hard copy due to marginal legibility of original document.] (Author/SP)

ED037097

PROCESS WITH MICROFICHE AND
PUBLISHER'S PRICES. MICRO-
FICHE REPRODUCTION ONLY.



FINAL REPORT

OEG 8-9-080032-2013 (057)

RESEARCH, DEVEL ND V
OF
DAILY DEMAND R

97

Research, Development and Validation of the Daily Demand Computer Schedule 360/50

By

Glen F. Ovard
and
Vernon C. Rowley

EM 007 700

EDUCATION EXPERIMENTAL
PROGRAMS
AND
LABORATORY SCHOOLS

COLLEGE OF EDUCATION
BRIGHAM YOUNG UNIVERSITY
PROVO, UTAH

This document was processed for the ERIC Document Reproduction Service by the ERIC Clearinghouse at Stanford. We are aware that some pages probably will not be readable in microfiche. However, this is the best available copy, and we feel that the document should not be withheld from interested readers on the basis of these unreadable pages alone.

PROCESS WITH MICROFICHE AND
PUBLISHER'S PRICES. MICRO-
FICHE REPRODUCTION ONLY.

Final Report

Project No.: 8-H-032
Grant No. OEG 8-9-080032-2013 (057)

ED037097

Research, Development and Validation of the Daily Demand Computer Schedule 360/50

Glen F. Ovard
and
Vernon C. Rowley
Brigham Young University
Provo, Utah

February 15, 1970

The research reported herein was performed pursuant to a contract with the Office of Education, U.S. Department of Health, Education, and Welfare. Contractors undertaking such projects under government sponsorship are encouraged to express freely their professional judgment in the conduct of the project. Points of view or opinions stated do not, therefore, necessarily represent official Office of Education position or policy.

Research, Development and Validation
of the
Daily Demand Computer Schedule 360/50

By

Glen F. Ovard
and
Vernon C. Rowley

U.S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE

Office of Education
Bureau of Research

U.S. DEPARTMENT OF HEALTH, EDUCATION
& WELFARE
OFFICE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

PART I. PRELIMINARY PAGES

Title Page

Table of Contents

Preface and Acknowledgements

TABLE OF CONTENTS

PART I. PRELIMINARY PAGES ii

Title Page. i
Table of Contents iii
Preface and Acknowledgements. iv

PART II. INTRODUCTORY SECTION. 1

A. Summary 2
B. The Problem 5
C. Characteristics of the Daily Demand Computer Schedule. 6
D. Terminal Objectives 7
E. Procedure and Methods 8

PART III. REVIEW OF LITERATURE. 11

PART IV. FINDINGS AND ANALYSIS 22

A. Creation of 360/50 Daily Demand Computer Schedule . . . 23
B. Validation of the DDCS 360/50 Program 37
C. Research and Testing of the 360/50 Program at a Selected Pilot School 41

PART V. CONCLUSIONS AND RECOMMENDATIONS 46

A. Conclusions 47
B. Recommendations 48

PART VI. SUPPLEMENTARY AND APPENDIX MATERIALS. 49

A. References. 50
B Bibliography. 51
C. Glossary. 56
D. Supplementary and Appendix Materials . . . 58
Appendix A. 59
Appendix B. 67
Appendix C. 233

Preface and Acknowledgements

Interest in scheduling students on a daily basis had its origin in the individualized continuous progress instructional system developed at the Brigham Young University Laboratory School. It was recognized early that traditional scheduling procedures placed some rather severe restrictions upon the individualized progress of many students, in that it encouraged a "lock-step" type of education. It was felt that the solution would be the development of a scheduling system that would recognize individual differences.

The staff and administration of the laboratory school, along with the Education Experimental Programs, created the first successful Daily Demand Computer Scheduling System, utilizing the IBM 7040-Q4 Computer. This system was piloted for fourteen days during 1965 and placed in operation for forty consecutive days during 1966. The initial scheduling system was designed essentially for the unique programs at the laboratory school. The next logical step would be to research, develop, and validate an expanded scheduling system to accommodate the enrollment and scheduling requirements of a large metropolitan high school.

It would require a third generation computer for a program of this magnitude. The IBM 360/50 system was selected as the testing and validating medium.

We would like to acknowledge the extensive assistance and cooperation of the BYU Computer Center, Gary Carlson, director. The following programmers and systems analysts made significant contributions: Matthew Wheaton, Glen Hammer, Robert Quist, Robert Rowe, Bruce Nielson, and Steven Howes in this regard.

Among those individuals involved in the educational design were Glen F. Ovard, M. Rex Arnett, Lowell Thompson, Wallace Allred, and members of the faculty of Brigham Young University Laboratory School.

We would also like to express our appreciation to the Clark County School District; John McCleary, director of the computer center; and the faculty and staff of Valley High School, Las Vegas, Nevada for assisting us in obtaining data used in this report.

The development of the Daily Demand Computer Schedule required the cooperative efforts of a large number of people. It is to be hoped that this is a major step forward in making time a tool for learning.

PART II. INTRODUCTORY SECTION

A. Summary

B. Problem

C. Characteristics of the Daily Demand Computer Schedule

D. Objectives

E. Procedure and Methods

Research, Development and Validation of the Daily Demand Computer Schedule 360/50

A. Summary

The Statement of the Problem

The purpose of this study is to further the research, development and validation of the Daily Demand Computer Schedule (DDCS).

A system by which students can be rescheduled daily for facilitating their individual continuous progress through the curriculum, which will allow teachers to regroup students as needed based upon that progress, and which will make time a tool of the teacher and learner, has been under development at the Brigham Young University for five years. The first computer schedule in the nation which would provide for daily rescheduling of students was completed by the Education Experimental Programs and Laboratory schools of the Brigham Young University in 1965. This Daily Demand Computer Schedule (DDCS) was in use 14 days at the end of the school year. During 1966 the program was rewritten as a result of this micro-study. In 1966 the new program was in operation 40 consecutive school days. During 1967 the research data obtained from this 40 operating days was compiled and interpreted. Based upon this data it was concluded that further research, development, testing and validation in a large urban pilot school outside of the Brigham Young University Laboratory School was needed. A third generation computer, such as IBM 360/50, would be required to adequately test the enlarged parameters of this new program.

Objectives

The specific terminal objectives of this study and for which funding was received are as follows:

1. To redesign and write the DDCS 7040 computer program for a 360/50 computer. The redesign would
    a. Generalize the current computer scheduling program from a specific schedule to one which could be used with any size of school and under varying circumstances.
    b. Include changes for improvement of the program which were found as a result of forty days of continuous operation on the DDCS 7040 program.
    c. Provide greater efficiency in operation through use of a larger computer.
2. Validate the new computer schedule with the input data previously used in the forty days of consecutive operation on the 7040 program.

Procedure

The 360/50 computer is a "third generation" computer having greater capacity for storage and data processing than the 7040. To maximize the use of this computer and to accomplish Objective No. 1, the following procedures were followed:

1. Establish a preliminary plan for management.
2. Hire systems analyst and programmers.
3. Review with them all reports, data in files and 7040 computer program.
4. Determine additional educational design changes which are needed to improve the current program, based on the 40 days of operation.
5. Determine new parameters required to generalize the current program to schools of varying size and under different circumstances.
6. Make a systems analysis of the new program.
7. Write the 360 computer program.
8. De-bug the program.
9. Validate the new program with input data previously used during the 40 days of consecutive operation during 1966 (This input data is presently being stored on tape).
10. Select a school or schools for research, testing and validation of the 360 computer program.
11. Using actual data from the pilot school, research and test the program. (The school will provide the data as if they were on an operating schedule. The schedules will be made and analyzed for possible revision.)
12. Modify program as required.
13. Summarize findings and conclusions.
14. Prepare final report for distribution through the U.S. Office of Education and the ERIC System.

Findings

1. The 360/50 Daily Demand Computer Scheduling System has successfully produced complete student schedules, teacher schedules, and class lists utilizing both the original Brigham Young Laboratory School data and data obtained from the selected pilot school.
2. The IBM 360/50 program requires additional testing in an operational school to attain optimum efficiency.

Conclusions

1. The Daily Demand Computer System is a viable tool that has real implications in extending the scope of individualized instruction.
2. Any school that has an individualized learning program and has access to an IBM 350 computer can now move to a Daily Demand Computer Schedule.

Recommendations

1. It is recommended that the process of testing, development, and modification of the Daily Demand Computer Schedule be continued.
2. It is recommended that any school that has IBM 360 computer availability and is concurrently involved in an individualized continuous progress educational program consider the possibility of utilizing the Daily Demand Computer Scheduling System.

B. The Problem

The purpose of this study was to further the research, development, validation, and generalization of the Daily Demand Computer Schedule (DDCS).

The Daily Demand Computer Schedule is a system of instructional management whereby students can be rescheduled daily to facilitate their individual continuous progress through the curriculum. This system permits teachers to regroup students as needed and to make time a tool of the teacher and learner.

The initial computerized daily scheduling program was completed by the Education Experimental Programs and Laboratory Schools of the Brigham Young University in 1965. This Daily Demand Computer Schedule was used as a pilot study for fourteen days at that time. The program was revised and placed in operation for forty consecutive days during 1966.

The research data obtained from this period of operation has been compiled and interpreted. Based upon this data it was concluded that further research, development, testing, and validation in pilot schools outside the Brigham Young University Laboratory School was needed. In addition, it was necessary to redesign and rewrite the program for the IBM 360/50 computer and to expand the parameters to accommodate the enrollment of large metropolitan high schools.

C. Characteristics of the Daily Demand Computer Schedule

The Daily Demand Computer Schedule has the following desirable characteristics:

1. Time becomes a tool for instruction and learning to be managed daily dependent upon the student's immediate needs.

2. Students are rescheduled daily based upon their individual progress in each curriculum area. A new master schedule is generated daily.

3. Time for learning is modular in that varying amounts of time can be grouped for specific tasks, while other tasks can be assigned completely different time allotments.

4. Independent study time is provided in varying amounts of time each day, with students determining how that time is spent in learning situations.

5. Provision is made for all types of teaching-learning strategies: large-group, small-group, individualized study, open laboratories, etc.

6. Students can be regrouped daily if necessary, based on common needs of individuals as they progress through the curriculum. There are no limitations on the type of groups that can be formed or how frequently these groups can be changed in size or composition.

7. Each student's daily schedule is completely individualized based on the needs of the student as determined by the teacher and the student.

8. The staff, professional and paraprofessional, can be scheduled in teams, as individuals, or in any specific combination that will enhance the learning situation.

9. Time during the school day can be scheduled for such things as student activities, rehearsals, field trips, practices, assemblies, and counseling appointments without disrupting regular class work.

10. Greater responsibility for learning is assumed by the individual student as he makes time a tool of learning.

D. Terminal Objectives

The objectives of this study are stated as terminal objectives. These objectives, with accompanying procedures to be followed, are indicated below:

1. To redesign and write the DDCS 7040 computer program for a 360 computer. The redesign will:
    a. Generalize the current program from a specific schedule to one which can be used with any size of school and under varying circumstances.
    b. Include changes for improvement of the program which were identified as a result of forty days of continuous operation on the DDCS.
    c. Provide greater efficiency in operation through use of a larger computer.
2. Validate the new computer schedule with the input data previously used in the forty days of consecutive operation on the 7040 program.
3. Research, test, and validate the new computer program in one or more pilot schools out in the field.

E. Procedure and Methods

To show how the specific objectives were accomplished, the methods and procedures utilized are given with each objective.

Objective No. 1

To redesign and write the DDCS 7040 computer program for a 360/50 computer. The redesign will

1. Generalize the current computer scheduling program from a specific schedule to one which can be used with any size of school and under varying circumstances.

2. Include changes for improvement of the program which were found as a result of forty days of continuous operation on the DDCS.

3. Provide greater efficiency in operation through use of a larger computer.

Procedure

To accomplish Objective No. 1, the following basic procedures and methods were utilized:

1. A systematic management efficiency strategy was evolved.

2. System analysts and programmers were hired.

3. The basic educational design was reviewed with the analysts and programmers.

4. All previous reports, file data and the 7040 computer program were reviewed with all staff members.

5. The new parameters required to generalize the current program to schools of varying size and under different circumstances were determined.

6. The system analysis for the new program was made appropriate. Appropriate flow charts were made to assist the programmers.

7. The computer program was written in Fortran IV.

8. The program was subjected to the de-bugging process (This process is on-going and must be continued as the program is tried in the different schools and under various circumstances).

9. The various parts of the program were tested, using real data.

10. A tentative program was declared operational.

Objective No. 2

Validate the new computer schedule with the input data previously used in the forty days of consecutive operation on the 7040 program.

Procedure

1. In 1966 the Brigham Young Laboratory School was on the DDCS 7040 computer schedule for forty consecutive days. The requests for student time made by the teacher, the master schedule generated by the computer, and the individual student schedules were stored on tape. The teacher requests on selected days were submitted to the new 360/50 computer scheduling program.
2. De-bugging of the 360/50 computer schedule using this stored data was continued.
3. Completed student schedules were finally made using the 360/50 program.
4. The new student schedules were analyzed for completeness and efficiency.
5. Additions to the new scheduling program were recommended.
6. The new additions were programmed.
7. The program was tested again against the stored data.
8. The new output data (student schedules) was analyzed and conclusions made.

Objective No. 3

Research, test and validate the new computer program in one or more pilot schools out in the field.

Procedure

Using the revised 360/50 computer schedule, the following procedures were utilized:

1. Valley High School at Las Vegas, Nevada was selected as the pilot school.
2. A research director, a systems analyst, and an educational design consultant went to Valley High School to prepare and secure the following basic master file data which consisted of:

   Schedule of rooms
   Teacher files
   Course files
   Student files

3. The master file data was then punched on IBM cards for use with the 360/50 scheduling program.

4. Teachers and administrators of Valley High School submitted requests for five separate days of school operation.

5. The requests were edited for errors by our scheduling coordinator.

6. The requests as input data were then submitted to the computer and scheduled on the 360/50 computer.

7. Three separate master schedules were generated by the computer for each of three days of requests.

8. Student schedules, teacher schedules and class lists were printed.

9. These were analyzed for completeness and efficiency.

10. De-bugging of the scheduling program using the Las Vegas data was commenced.

11. The programs were modified and improved.

12. The input data was rescheduled utilizing the revised program.

13. The final report was compiled and forwarded to the U.S. Office of Education and ERIC Centers for dissemination purposes.

PART III. REVIEW OF LITERATURE

Part III. Review of Literature

Computerized scheduling can mean many different things. For some it may imply the preparation of a conflict matrix prior to the construction of a schedule. To others it may indicate the actual computerized examiniation of the variables against a system of established parameters to prepare the organizational framework of the school, the master schedule, as well as individual schedules for each student.[1]

If the school is operating on a traditional schedule where the instructional framework consists of time patterns and group sizes that are relatively fixed and constant, the computerized schedule is simply a system of handling the logistics of scheduling in a more efficient manner. The instructional pattern in this situation would not be affected by the use of the computer for scheduling purposes. The preparation of this type of schedule by computer is relatively simple and inexpensive compared to building the master schedule for a large secondary school by hand. This is one of the many areas on which computerized data processing can aid in eliminating time - consuming clerical tasks and on providing more accurate and legible educational information.[2]

The computerized flexible schedule is a computerized instructional framework that is designed to permit variable time patterns, variable group sizes, team teaching, and student-centered curriculum in an attempt to individualize instruction and learning. According to Duncan:

> Flexible scheduling serves as a departure from the historical assumptions underlying learning experiences . . . In this form, the innovation moves toward a closer harmony with the body of psychological data which has been uncovered regarding learning. Flexible scheduling attempts to place the schools one step closer to the scientific principles of learning which educators have uttered for scores of years.[3]

Several patterns of computerized flexible scheduling have been developed. The Stanford School Scheduling System (SSSS) developed at Stanford University[4], the Generalized Academic Simulation Program (GASP) developed at Massachusetts Institute of Technology[5], The Indiana Flexible Schedule (IndiFlexS) developed at the University of Indiana[6], and the Daily Demand Computer Schedule (DDCS) developed at Brigham Young Laboratory School[7] are significant examples. The latter two systems are briefly described as follows:

---

[1] Duane Richardson, "Pupil Scheduling by Computer," Educational Data Processing: New Dimensions and Prospects, ed. Richard A. Kaimann and Robert W. Marker (Boston: Houghton Mifflin Company, 1967), p. 190.

[2] B. H. Wulff, "Data Processing for Student Scheduling: An IBM Approach," Journal of Secondary Education, 36:380-81, October, 1961, p. 381.

[3] John Richard Duncan, "A Study of Teachers' Expectations in Three Selected Secondary Schools" (unpublished doctor's dissertation, Indiana University, 1967), p. 9.

[4] Robert N. Bush and Dwight W. Allen, A New Design For High School Education (New York: McGraw-Hill, 1964).

The Stanford School Scheduling System

The Stanford system of computerized flexible scheduling provides a single master schedule on an academic year, or more rarely, a semester basis. It has usually been necessary to determine course offerings and preregister students early in the second semester of the previous school year in order to provide the input information necessary for the initial construction of the schedule. The system provides for the exercise of a considerable amount of choice in the selection of the operating variables.[8]

The basic provisions include the ability to organize a given course of study into various components utilizing various modules of time, students, and space. The system provides for the selection of a cycle which may range from one to ten days in length, but when once selected is constant for all courses for the duration of the schedule. If a traditional type of schedule is selected, the cycle will be a single day, as the schedule would repeat itself each day. The most commonly selected cycle variation of the Stanford School Scheduling System is the five-day cycle in which all Mondays are alike, all Tuesdays are alike, etc.; but in which the schedule for Tuesday (or any other day) may have little relationship to the schedule for Monday (or any other day).

Once the cycle has been established, the next step is to select the length and number of time modules in a given day. These may vary from ten to sixty minutes. The most commonly selected modules are fifteen, twenty, or thirty minutes in length. Each course of study is then authorized a minimum number of modules per week (or per cycle) that must be scheduled for each student to meet such requirements as accreditation standards. The teachers then have the opportunity to use these parameters to develop a course structure that will most effectively meet the needs of the course. According to Trump and Miller

---

[5] Judith Murphy, School Scheduling by Computer--The Story of GASP (New York: Educational Facilities Laboratories, 1964).

[6] Donald C. Manlove and David W. Beggs, Flexible Scheduling: Bold New Venture (Bloomington, Indiana: Indiana University Press, 1965).

[7] Glen F. Ovard and M. Rex Arnett, "Daily Demand Computer Scheduling (DDCS)--A New Unique Scheduling Program," Change and Secondary School Administration, ed. Glen F. Ovard (New York: The Macmillan Company, 1968), pp. 346-352.

[8] Robert V. Oakford, Dwight W. Allen, and Lynn Chatterton, "School Scheduling--Practice and Theory," Journal of Educational Data Processing, 4:16-50, Winter, 1966-67, p. 22.

The goal of the schedule is to give teachers and pupils as much freedom as is reasonable in the use of time, space, and numbers of persons, as well as content for instruction. The following are worthwhile objectives:

1. The class schedule may be changed daily, or at least frequently on the basis of teacher requests.

2. Each pupil, under competent direction and with appropriate controls, may make choices regarding his part in the established schedule.

3. Conflicts for pupils and teachers are reduced to a minimum.

4. Teacher loads and pupil loads permit both maximum professionalization of teaching and maximum individualization of learning opportunities.

5. The school knows what its pupils and teachers are doing and where they are.

6. The whole scheme is financially feasible and logistically operational.[9]

The SSSS system permits a course to be trouped in several different ways. A common procedure is for the teacher to divide the course structure into at least three instructional phases: large group, laboratory, and small group instruction. The intent of the large group instruction is to provide a framework whereby teacher-centered instruction can be given to as large a group as possible, thus eliminating excessive repetition and conserving teacher time. The large group may meet only once or twice a week.

A second phase of instruction, laboratory instruction, may meet two or three times a week with a much smaller group of students. The time may be varied to permit a science lab to meet for 90 minutes, a shop to meet for 150 minutes, a social studies lab to meet for 60 minutes, or other selected variations.

The small group phase, or seminar, is utilized to permit a teacher to meet a small group of from eight to sixteen students on a regularly scheduled basis for discussion groups. Both teachers and student consider the seminar phase to be one of the true advantages of computerized flexible scheduling. A fourth (and occasionally overlooked) phase is that of independent study.

---

[9] J. Lloyd Trump and Delmas F. Miller, Secondary School Curriculm Improvement (Boston: Allyn and Bacon, 1968), p. 307-308.

The type of computerized flexible scheduling typified by the SSSS system has been operated successfully in more than one hundred secondary schools in the last five years.

According to Van Dusseldorp and Richardson, "Both the Stanford and GASP programs have been used successfully to schedule secondary schools. However, a great deal of work still needs to be done in the development of better schedule-building programs."[10] The next section describes some of the attempts to meet this need and, at the same time, provide a better system of education for secondary school children.

The Daily Demand Computer Scheduling System

The Daily Demand Computer Scheduling System (DDCS) as developed at Brigham Young Laboratory School can be defined as a system by which appropriate instructional staff members determine what classes are taught, how many sections will be offered, for how long they will be taught, who will attend them, who will be responsible for conducting the learning activities, and in what type of grouping arrangements the teaching will occur on a daily basis. The term "daily demand" is significant in that the scheduling is done daily in response to the constantly changing demands or needs of students, teachers, and other personnel.

The most obvious difference between DDCS and other types of flexible scheduling is the frequency of the scheduling cycle. The reason for their difference is most significant in terms of individual students progressing at their own individual rates. The daily rescheduling is done on "demand" to provide the exact amount of instructional time, space, and grouping that is most pertinent to each individual student at that particular stage on his educational development.[11]

According to Arnett, there are two particular capabilities inherent in DDCS which are not present in other flexible scheduling systems:

> First, there is more flexibility to schedule according to actual instructional needs since no teacher is tied down to a schedule made a semester or more in advance. Because there need not be a fixed weekly scheduling pattern, the time for each subject area can be varied to provide an arrangement which most nearly fits a particular learning objective.
>
> Second, as the amount of time is flexible, so is the grouping of students who are scheduled for instruction. Students may be shifted from one group to another as their individual needs change. At any time that it becomes apparent that a student does not seem to be receiving maximum benefit from a particular grouping situation, he is not obliged to continue in such a group throughout the semester. Making the change is not an imposition of the

---

[10] Ralph A. Van Dusseldorp and Duane E. Richardson, "Class Scheduling by Computer," Journal of Educational Data Processing, 4:51-60, Winter, 1966-67, p. 58.

[11] M. Rex Arnett, Daily Demand Computer Scheduling: Questions and Answers From a Student's Viewpoint (Provo, Utah: Education Experimental Programs, 1967), p. 2.

counseling staff as it can be made very readily by the teacher as part of the normal input data for a given day's processing.[12]

In the actual operation of the DDCS program, ordinary scheduling requests were made two days in advance of the day of use. Requests are made by physically assembling prekeypunched data processing cards into a "request deck." All requests together comprised the master request deck from which the daily master schedule was prepared by computer processing. Processing was completed the same day, and the output was printed and distributed to teachers the next morning. The basic planning was accomplished prior to the preparation of the request decks. By knowing the schedule at least a day in advance, teachers are able to do supplementary planning and set up the necessary equipment for laboratories, etc. Students also need to have this information in advance in order to have assignments completed at the proper time and to plan independent study activities.[13]

In addition to student and teacher schedules, rolls for all computer-scheduled activities are distributed to the proper departments or individual teachers.

Certain types of requests, such as those for an unusually large group, or for students not ordinarily scheduled for a course must be scheduled through a coordinator.

Early attempts at daily demand scheduling. One of the early attempts at daily demand scheduling occured at Brookhurst Junior High School in Anaheim, California.[14] Although the experiment was considered relatively successful, one of the major difficulties was the operation of the rescheduling function manually, using notched cards.

The computer generation of a daily demand scheduling system was first attempted at Brigham Young Laboratory School in April, 1966. The system was placed in operation for fifteen days at that time and sufficient information was obtained to provide a background for revisions which permitted a general implementation of the system beginning February 21, 1967.[15]

From a comparison of the results of these two early attempts at daily demand scheduling it appears that, in order to manage the complex task of scheduling such a variety of demands for maximum efficiency, it is necessary to employ the techniques of electronic data processing.

---

[12] M. Rex Arnett, "A Manuscript on Daily Demand Computer Scheduling at BYU Laboratory School," (unpublished manuscript, Stanford University, 1967), p. 6.

[13] Glen F. Ovard, and Others, Operational Procedure, Program Control Manuals, Program Flow Charts for Daily Demand Computer Scheduling (Provo, Utah: Education Experimental Programs, 1967), p. 3-4.

[14] Gardner Swenson and Donald Keys, Providing for Flexibility in Scheduling and Instruction (Englewood Cliffs, New Jersey: Prentice-Hall, 1966).

[15] M. Rex Arnett, "A Report to the Parent-Teachers Association on the Daily Demand Computer Scheduling Project" (Provo, Utah: Brigham Young University Laboratory School, 1967), p. 6.

The purpose of daily demand scheduling. The entire concept of daily demand scheduling is predicated upon individualizing instruction and learning.[16] The concept of individualized continuous progress curriculum began to be implemented at Brigham Young Laboratory School, under the auspices of the Education Experimental Programs, beginning in the 1959-1960 school year. The programs of continuous progress education had emphasized individualized learning, nongraded study, small group instruction, instructional materials centers, and continuous program testing and record keeping. Flexible scheduling on a daily basis through the use of a computer appeared to be a tool that would permit more effective utilization of these innovative educational patterns.[17]

Glen Heathers attempts to relate the new organizational patterns (which include various types of flexible scheduling) to individual differences among students in the following statement:

> . . . the new organizational plans bear important relationships to some of the societal forces and ideas that are reshaping American education. The clearest relationships are to the influences calling for better ways of dealing with differences among learners, for improving teaching of science and mathematics, and for increasing the efficiency with which our short supply of well-qualified teachers is being employed.[18]

Robert N. Bush appears to go a step farther when he encourages increasing experimentation in flexible scheduling:

> With regard to flexible scheduling, I have been encouraged that we are opening doors, but that we remain too timid in our approach . . . . There is also danger lest we adopt a flexible schedule and make everybody fit into it. There is little advantage in throwing off one set of orthodoxies for another. . .[19]

---

[16] Glen F. Ovard, Overview of Eight Years of Development in Continuous Progress Education (Provo, Utah: Education Experimental Programs, 1967), p. 7.

[17] Glen F. Ovard (ed.), Summary: Background, Basic Premises and Objectives of Continuous Progress Programs (Provo, Utah: Education Experimental Programs 1966), p. 1-5.

[18] Glen Heathers, "School Organization: Nongrading, Dual Progress, and Team Teaching," The Changing American School, Sixty-fifth Yearbook of the National Society for the Study of Education, Part II (Chicago: University of Chicago Press, 1966), p. 125.

[19] Robert N. Bush, "A Searching Appraisal of New Developments," Journal of Secondary Education, 37:321-326, October, 1962, p. 324.

Perhaps approaches such as the Daily Demand Computer Scheduling System can be effective in eliminating the rigidity and rescheduling problems inherent in earlier attempts at computer scheduling. There is also a possibility that the problem of conflicts that has plagued these earlier systems can be substantially alleviated.[20]

Evaluation of Flexible Scheduling Programs

A considerable amount of discussion has taken place concerning the relative merits of many of the recent forms of educational innovation, such as the computerized flexible schedule. Unfortunately the vast majority of the literature is completely subjective in nature. Opinions range from high enthusiasm through qualified acceptance to outright rejection. Noted educators have not hesitated to establish a position at one extreme or the other without seeming to require the scientific evidence to substantiate their stand.

Robert N. Bush and Dwight W. Allen present a strong circumstantial case for this type of instructional management.[21] However, evaluation of the effectiveness of their program is basically limited to such descriptive phrases as "Libraries are full of students," "Discipline problems are down," and "General academic pace is quickened."[22]

On the other hand, B. Frank Brown of Melbourne High School in Florida is equally certain that modular or flexible scheduling is nothing but a "gold-plated fad."[23]

The literature available concerning the effectiveness of curriculum modification as it applies to the flexible schedule can easily be separated into the areas of opinion and research. It seems appropriate to examine these points of view.

Opinion. Many opinions are formulated as a result of observations or direct involvement in a school that is operating with a flexible schedule. A report of the social studies instructional program at Homestead High School in Cupertino, California is an example of this type of evaluation.[24]

---

[20] Allen J. Gruman, "A Comparison of Computer Programs for the Assignment of Students to Classes: Report of an Experimental Study," Journal of Educational Data Processing, 4:7-15, Winter, 1966-67, p. 9-10.

[21] Robert N. Bush and Dwight W. Allen, A New Design for High School Education (New York: McGraw-Hill, 1964).

[22] Ibid., p. 186.

[23] Clayton Braddock, "Changing Times are Changing Schools," Southern Education Report, 3:24-30, October, 1967, p. 25.

[24] Jack A. Hamilton and Robert F. Madgic, "Can Flexible Scheduling Improve Social Studies Instruction?" Journal of Secondary Education, 39: 295-298, November, 1964.

In this article, the authors discuss some of the positive and negative aspects of their program. They particularly emphasized the value of the small group or seminar phase of instruction. Listed as positive outcomes were such factors as increased responsibility for students, encourages thinking, encourage alternate solutions, etc. It should be noted that no objective comparisons were made as to the amount of learning that actually took place. It should also be noted that in articles of this nature some authors will refer to specific situations while others simply express a feeling or insight.

The report of modular scheduling as it occurs in Memphis and other southern cities is very similar in nature. There does, however, seem to be more concern over the student's attitude and his perception of his role as a participant in an innovative learning situation.[25]

Decatur-Lakeview High School in Illinois was one of the first schools to publish an organizational and structural monograph of a modular schedule.[26] As part of their findings, they published a student attitudinal survey which reflected student opinion as to the comparative quality of modular system of education. This student opinion survey indicated that the ninth, tenth, and twelfth grades expressed generally positive attitudes, while the eleventh grade was extremely negative in almost all respects.[27]

The author of the Decatur-Lakeview report did not apologize for the lack of research findings. He strongly recommended that researchers translate theoretical findings into instructional practice.

A few authors have referred briefly to student achievement. E. Dale Davis stated that, ". . . schedule modifications have produced results in student achievement which are just as good--and, in many cases, better than--those of the traditional approaches."[28] He went on to say that student morale is better; teachers spend more time in preparation; teachers are freed from many nonprofessional tasks; more teaching aids are used; and specialists and teachers with special abilities are used to better advantage. Certainly opinions of this nature encourage the innovator and would-be innovator. They would be much more meaningful if substantiated by research.

In actual practice there is a tendency for the innovative school to begin a new curriculum modification before even the preliminary results are evaluated on the previous modification. Philip G. Kapfer discussed an instructional management strategy at Valley High School using an instructional package approach.[29] This program was actually implemented on a limited scale before the original innovation (modular schedule) was completely operational.

---

[25] Braddock, op. cit., 24-30.

[26] David W. Beggs, III, Decatur-Lakeview High School (Englewood Cliffs, New Jersey: Prentice-Hall, 1964).

[27] Ibid., p. 245-247.

[28] E. Dale Davis, Focus on Secondary Education (Glenview, Illinois: Scott Foresman and Company, 1966).

Research. It has been stated that the history of research as it applies to the effectiveness of flexible or modular scheduling is somewhat limited. A research proposal has been designed at Homewood-Flossmoor High School, Flossmoor, Illinois, to attempt to evaluate a school that is changing over from a traditional program to a flexible schedule.[30] The Homewood-Flossmoor research is based upon a time-series design because of the difficulty of obtaining a comparable control school. The designers indicated that it was not practical to attempt to operate a traditional program and a flexible schedule in the same school simultaneously.

The research at Homewood-Flossmoor was supported by the State of Illinois Department of Program Development for Gifted Children. The school has offered to supply the research plan to any school that will, in turn, share the results of its research.[31] The author of the Homewood-Flossmoor report indicated that, although a few school have used flexible scheduling for several years, little data have been reported on the effects of such a change. The available reports were basically informal.

An unpublished dissertation by Gerald Paul Speckhard provided additional insight into the need for further research.[32] According to his report, both students and teachers approved of the change to a flexible schedule in the experimental school, and both groups felt that the students learned as well or better than in the previous traditional system. The test results indicated that experimental students showed a greater ability in critical thinking, while neither were superior in study habits or attitude. Speckhard emphasized the difficulty in finding a control school similar to the experimental school.

A research study comparing English achievement of seventy-four ninth-grade pupils enrolled in a three-period team-teaching class with English achievement of 149 ninth-grade pupils enrolled in a conventionally taught class was reported in the literature.[33] Comparison of achievement revealed significant differences (at the .05 level) in the means for the experimental group vs. the control group on teacher-made tests and on the California Reading Test. No significant interactions were revealed. It was concluded that the findings were sufficiently positive to warrant further implementation and evaluation of the program.

---

[29] Philip G. Kapfer, "An Instructional Management Strategy for Individualized Learning," Phi Delta Kappan, 49:260-263, January, 1968.

[30] Theodore R. Storlie, "Evaluating Flexible Scheduling," Educational Leadership, 25:177-183, November, 1967.

[31] Ibid., p. 298.

[32] Gerald Paul Speckhard, "An Evaluation of the Educational Program of a High School Using a Modular Schedule," (unpublished doctor's dissertation, University of Colorado, Boulder, 1966).

[33] William Georgiades and Joan Bjelke, "Evaluation of English Achievement in a Ninth Grade, Three-Period Team-Teaching Class," California Journal of Educational Research, 17:100-112, May, 1966.

One of the more recent, and perhaps more significant, studies was recently completed by L. Leon Webb. This study examined some of the transitional effects that occurred when Brigham Young Laboratory School (which was using the original DDCS program) was closed and the students were transferred to the more traditional schools in the local area.[34]

Webb concluded that students who have been involved in an individualized, continuous progress program such as that facilitated by the Daily Demand Computer System at Brigham Young Laboratory School will do as well or better academically than students who have not been involved in this type of program.[35] In addition there are a number of positive factors in regards to the students' attitudes and involvement that would encourage further examination and evaluation of these types of programs. For example: Students who had been involved in individualized, continuous progress programs express positive attitudes about various aspects of this type of programming; the program more closely fits students' professed individual needs; involvement in this type of program is related to student preference for the program, and the traditional program is perceived by students as being less difficult, and less interesting than the individualized, continuous progress program.[36]

The beginning attempts at daily demand scheduling appear to have been a valuable aid to the individualization of instruction and learning. The development of more sophisticated systems, such as the Daily Demand Computer Schedule that has been generalized for third-generation computers, can perhaps do a great deal to more fully meet the needs of each individual student. The following statement made very early in the historical development of computerized schedules is coming increasingly closer to reality:

> The possibility of developing a flexible high school schedule to serve rather than stifle the educational needs of pupils comes closer to reality with the advent of electronic data processing procedures and giant high-speed computers . . . The application of these modern procedures to high school scheduling appears most promising.[37]

---

[34] L. Leon Webb, "Selected Transitional Effects for Students Transferring from a Continuous Progress Program to a Non-Continuous Progress Program" (unpublished doctor's dissertation, Brigham Young University, 1969).

[35] Ibid., p. 214.

[36] Ibid., pp. 213-218.

[37] Robert N. Bush and Dwight W. Allen, "Flexible Scheduling for What?" Journal of Secondary Education, 36:346-353, October, 1969, p. 349.

PART IV. FINDINGS AND ANALYSIS

A. Creation of 360/50 Daily Demand Computer Schedule

B. Validation of the DDCS 360/50 Program

C. Research and Testing of the 360/50 Program at a Selected Pilot School

PART IV. FINDINGS AND ANALYSIS

The findings and analysis as presented in this section are related directly to the terminal objectives set at the beginning of the project and reviewed in Part II. These objectives are centered around (1) the creation of a new daily demand computer schedule which can be utilized by any size school with varying circumstances; (2) the validation of the new schedule using previously collected data, and; (3) verification of the validated schedule through use of real data from a large pilot high school.

A. Creation of 360/50 Daily Demand Computer Schedule

A Brief Description of The Daily Demand Computer Scheduling System

The Daily Demand Computer Scheduling System (DDCS) as developed at Brigham Young Laboratory School can be defined as a system by which appropriate instructional staff members determine on a daily basis what classes are taught, who will attend them, who will be responsible for conducting the learning activities, and in what type of grouping arrangements the teaching will occur. The term "daily demand" is significant in that the scheduling is done daily in response to the constantly changing demands or needs of students, teachers, and other personnel. In order to manage the complex task of scheduling such a variety of demands for maximum efficiency, it is necessary to employ electronic data processing.

The most obvious difference between DDCS and other types of flexible scheduling is the frequency of the scheduling cycle. Related to this are two particular capabilities inherent in DDCS which are not present in other flexible scheduling systems.

First, there is more flexibility to scheduling according to actual instructional needs since no teacher is tied down to a schedule made a semester or more in advance. Because there need not be a fixed weekly scheduling pattern, the time for each subject area can be varied to provide an arrangement which more nearly fits a particular learning objective.

Second, as the amount of time is flexible, so is the grouping of students who are scheduled for instruction. Students may be shifted from one group to another as their individual needs change. At any time that it becomes apparent that a student does not seem to be receiving maximum benefit from a particular grouping, he may be readily rescheduled into a more effective learning situation. Making the change is not an imposition on the counseling staff as it can be made very readily by the teacher as part of the normal input data for a given day's processing.

In Daily Demand Computer Scheduling, teachers submit requests for students to be scheduled. They may request any or all of the students enrolled for a particular course, and several requests may be submitted for each course. Each request, besides specifying which students are wanted, also indicates how much time is desired. Further, a teacher may specify precisely when and where he wants the group to meet; but normally he will let the computer select a time and place so that fewer conflicts will be created with other teachers' requests.

The actual requests are prepared by assembling prekeypunched cards into a "request deck." As may be seen in Figure 1, these request decks are submitted through a coordinator who modifies requests which may cause irresolvable conflicts and approves special requests such as those for unusually long periods of time or those which must occur at a specified time. Teachers make these requests on Mondays for Wednesdays, Tuesdays for Thursdays, Wednesdays for Fridays, etc.

The requests are assembled and taken to the computer for processing. They are read onto magnetic tape. Then the actual scheduling takes place on the IBM 360/50. Final print-out provides individual student schedules, teachers schedules, corresponding class rolls, and administration reports.

On the following morning the schedules are distributed in homerooms, each of which is supervised by a teacher. Thus these schedules are seen by students and teachers a day in advance of their actual use so that class assignments can be prepared and students may be prepared for exams, discussions or other activities. Before the current day's schedules are picked up for use, students specify how they will utilize any unscheduled time. They may sign up for extra work in a laboratory or for study in an area where resources and assistance are available for pursuing depth study independently in a given subject area. Counseling appointements or consultation with teachers are also scheduled by students as they have need and according to the times that teachers and counselors are available.

Duplicate copies of the student schedule serve to permit location of a student in case of an emergency.

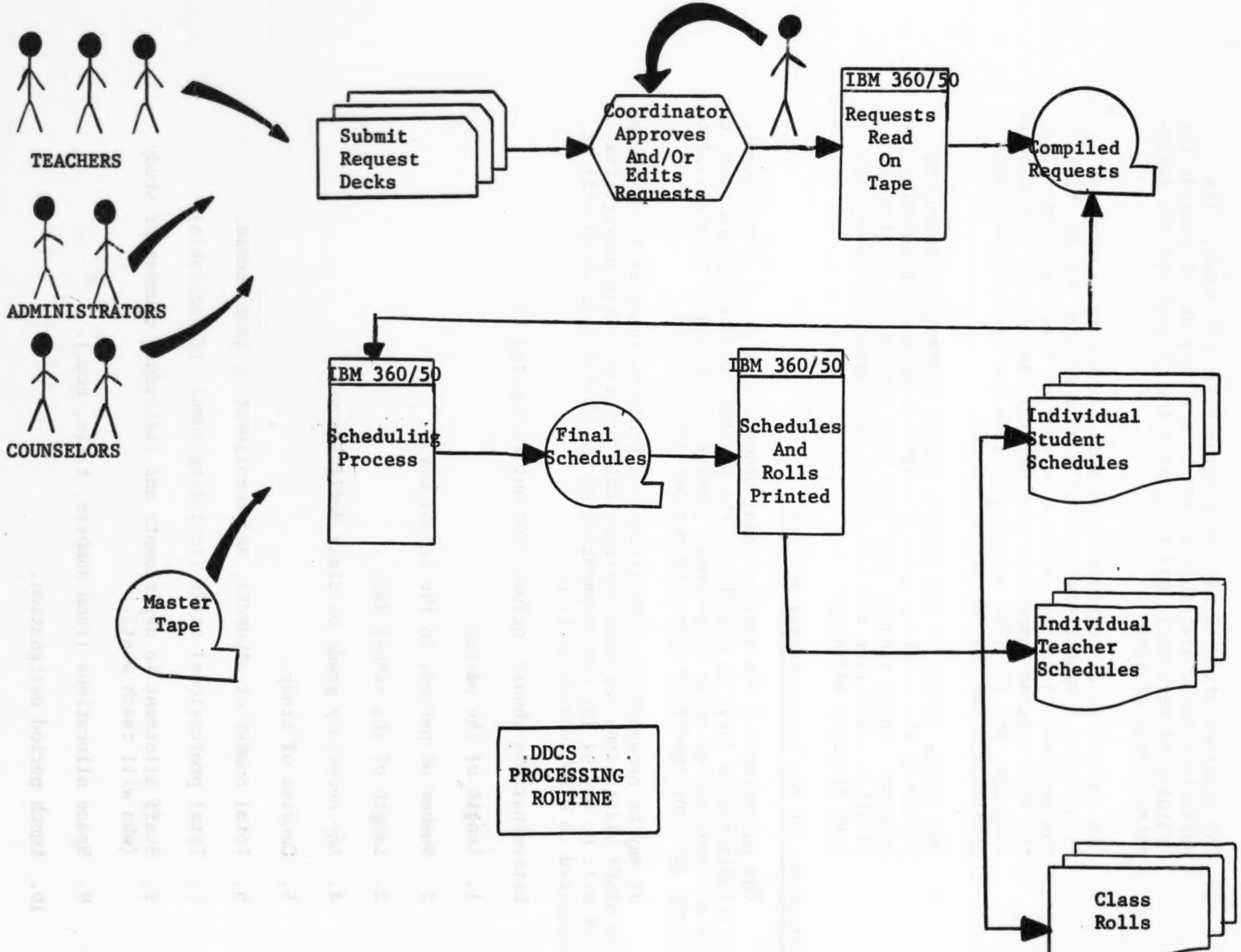
The flexibility in the overall schedule is to a great extent only limited by the teacher's exploitation of the system. While all of the specific possibilities are not enumerated, generally teachers have control over (1) length of class time, (2) frequency of class contact, (3) grouping, and (4) mode of instruction: open individualized laboratory vs. teacher-directed group activities.

Resource Considerations

Once the philosophy of individualized learning has been accepted and the decision has been made to examine the possibilities of utilizing Daily Demand Computer Scheduling to more fully accomplish this goal, it is necessary to examine the availability of resources that are required for successful implementation of the program.

It is almost axiomatic that no major innovation can be successfully undertaken without adequate resources and preparation. Some of the factors that should be considered prior to involvement in the DDCS program are as follows:

1. The availability of adequate preparation time is a major consideration. There must be sufficient time to provide for training of the staff, orientation of the community and prospective students and to prepare the necessary input data for the computer.


TEACHERS
ADMINISTRATORS
COUNSELORS
Submit Request Decks
Coordinator Approves And/Or Edits Requests
IBM 360/50
Requests Read On Tape
Compiled Requests
Master Tape
IBM 360/50
Scheduling Process
Final Schedules
IBM 360/50
Schedules And Rolls Printed
Individual Student Schedules
Individual Teacher Schedules
Class Rolls

DDCS PROCESSING ROUTINE

FIGURE 1

2. The implementation of any new program will cost money. Are sufficient funds available to set up the program, to provide for training of personnel, and to operate the program for the designated period of time.

3. The technological considerations must be thoroughly examined. Is adequate computer time available on a regular basis? Will the available hardware handle a program of this dimension? Is the DDCS program compatible with the computer system (Job-Control Language, etc.)? (The existing program has been prepared and generalized for the IBM 360/50 system).

4. The unique nature of the present school system will definitely influence the relative ease of establishing and implementing the program. Those schools that have been previously involved in flexible scheduling and individualized learning programs will have distinct advantage.

Parameters of the Instructional Program

The parameters of the instructional program at a given school should be established as early as possible. The present computer program provides for a school day of up to 28 periods. However, this can be generalized to accommodate any specified number of periods less than 28.

It may be necessary to preregister the students in order to determine the staff allocations for each instructional program. This preregistration will be useful for the preparation of the Master File which will be discussed in a subsequent section.

Determinations should include, but not be limited to

1. Length of the period.
2. Number of periods in the school day.
3. Length of the school day.
4. Any necessary grade or class designations.
5. Courses of study.
6. Total number of students, and enrollment in each course.
7. Total professional staff (including names, if available).
8. Staff allotment to departments and individual courses of study (who will teach what).
9. Space allocations (room numbers, title, etc.).
10. Lunch period designation.

Preparations of Master File Data

Careful preparation of the initial master file data cannot be overemphasized. It is recommended that all portions of this file be carefully audited before the data is actually inserted into the computer. Provisions have been made for correction and updating but these will not compensate for a casually prepared master file.

The purpose of the master file is to provide for a primary course structure (course, teachers, students, rooms) as well as a "reservoir" that can be drawn upon during the restructuring that occurs in the daily operations of this computer scheduling program.

The input cards for the master file preparation are basically the same cards that will be used to build course request decks in the actual operation of the program.

Each scheduling operation utilizing a different computer system and/or installation will require somewhat unique job-control language to insert the master file information and arrange for its recall.

The master file data should be designed as indicated and inserted in the following sequence:

1. (Grade File)
   11FT08F001 DD *
   GRADE cards
   /* (one card for each grade--12 cards maximum)

2. (Course Information File--Must contain all courses which may be requested)
   11FT08F002 DD *
   COURSE cards (for specific course)
   ROOM cards (0 to 10 rooms, in order of preference)
   TEACHER cards (0 to 10 teachers, all teachers who will be regularly scheduled for course)
   STUDENT cards (0-2,500 students, all students who are regularly scheduled for the course)
   (Repeat for each course--500 courses maximum)
   /*

3. (Course Name File)
   11FT08F003 DD *
   COURSE cards (One card for each course in ascending numberical order)
   /*

4. (Room Name File)
   11FT08F004 DD *
   ROOM cards (one card for each available room in ascending numerical order-500 rooms maximum)
   /*

5. (Teacher Name File)
   11FT08F005 DD *
   TEACHER cards (one card for each available teacher; including counselors, dean, administrators, etc.--500 teachers maximum)

6. (Student Name File)
11FT08F006 DD *
STUDENT cards (one card for each student enrolled in the school--2,500 students maximum)
/*

(End of master--follow by either execute card or // card.)

Control Card Preparation

The majority of the control cards are in frequent use and should be prekeypunched in sufficient quantities to prepare master file information as well as to provide personnel with the necessary cards to make requests appropriate to their own academic or administrative area. Thus the problem of keypunching a vast amount of input data daily is eliminated.

Control cards should be color coded to identify and distinguish the various types of cards. Standard IBM 1580 or 5081 data processing cards or plain data processing cards may be used. No custom printed or nonstock cards are necessary in the system. COURSE cards, however, should have the diagonal alignment cut located in the upper left corner. All other cards should be cut at the upper right corner. Thus the rounded corner of each COURSE card at the right side will provide a convenient divider for separating various requests in a file tray.

In preparing the cards it should be noted that the card title is left-justified and begins in column 1, additional information will be placed in the designated columns, and all numbers are right-justified (i.e., the numeral four in columns 79 and 80 would be punched as 04 with the "4" being in column 80).

Control cards should be punched in the following manner. All column designations are inclusive:

1. The following control cards will be used to build the master file. The GRADE card will be used in the master file only, while the COURSE, ROOM, STUDENT, and TEACHER cards will be used in the daily operations of the DDCS system. The GRADE card is used to designate the grade placement of students (where such designation is desirable).

GRADE Format:	GRADE in columns 1-5, grade number (-1-12) in columns 9-10, student number of first student in the sequence in columns 17-20, student number of last student in the sequence in columns 27-30, the card type number 52 in columns 79-80.
COURSE Format:	COURSE in columns 1-6, course title in columns 10-17, course number in columns 57-60, card type number 04 in columns 79-80.

ROOM Format:	ROOM in columns 1-4, room name in columns 10-17, room number in columns 57-60, card type number 03 in columns 79-80.
STUDENT Format:	STUDENT in columns 1-7, student name (last name first) in columns 10-33, student number in columns 57-60, card type number 01 in columns 79-80.
TEACHER Format:	TEACHER in columns 1-7, teacher name (last name first) in columns 10-33, teacher number in columns 57-60, card type number 02 in columns 79-80.

2. The following control cards will be used in conjunction with appropriate COURSE, ROOM, STUDENT, and TEACHER cards to initiate changes in the master file:

ADD Format:	ADD in columns 1-3, card type number 05 in columns 79-80.
CHANGE Format:	CHANGE in columns 1-6, card type number 06 in columns 79-80.
DELETE Format:	DELETE in columns 1-6, card type number 07 in columns 79-80.
PURGE Format:	PURGE in columns 1-5, card type number 53 in columns 79-80.

3. The following control cards will be used in conjunction with appropriate COURSE, ROOM, STUDENT, and TEACHER cards to prepare individual course request decks in the daily operation of the DDCS system:

PERIODS Format:	PERIODS in columns 1-7, number of periods (01-28) in columns 59-60, card type number 10 in columns 79-80.
GROUP Format:	GROUP in columns 1-5, number of groups (01-28) in columns 59-60, card type number 15 in columns 79-80.
OPEN Format:	OPEN in columns 1-4, number of periods (01-28) in columns 59-60, card type number 16 in columns 79-80.
MAY Format:	MAY in columns 1-3, number of MAY groups (01-28) in columns 59-60, card type number 17 in columns 79-80.
OPEN (MAY) Format:	OPEN(MAY) in columns 1-10, number of periods (01-28) in columns 59-60, card type number 18 in column 79-80.

NO PRIORITY Format:	NO PRIORITY in columns 1-11, card type number 19 in columns 79-80.
PRIORITY Format:	PRIORITY in columns 1-8, card type number 20 in columns 79-80.
ROOM NONE Format:	ROOM NONE in columns 1-9, 0000 in columns 57-60, card type number 13 in columns 79-80.
STUDENT ALL Format:	STUDENT ALL in columns 1-11, 9999 in columns 57-60, card type number 11 in columns 79-80.
STUDENT NONE Format:	STUDENT NONE in columns 1-12, 0000 in columns 57-60, card type number 12 in columns 79-80.
TEACHER NONE Format:	TEACHER NONE in columns 1-12, 0000 in columns 57-60, card type number 14 in columns 79-80.
STARTING PERIOD Format:	STARTING PERIOD in columns 1-15, starting period number (01-28) in columns 59-60, card type number 21 in columns 79-80.
BETWEEN PERIOD Format:	BETWEEN PERIOD in columns 1-14, period number (01-28) in column 59-60, card type number 22 in column 79-80.
AND PERIOD Format:	AND PERIOD in column 1-10, period number (01-28) in columns 59-60, card type number 23 in columns 79-80.
NO SHORT Format:	NO SHORT in columns 1-8, card type number 24 in columns 79-80.
NO INDIVIDUAL SHORT Format:	NO INDIVIDUAL SHORT in columns 1-19, card type number 25 in columns 79-80.
NO LATE SHORT Format:	NO LATE SHORT in columns 1-13, card type number 26 in columns 79-80.
CONFLICTS MAXIMUM Format:	CONFLICT MAXIMUM in columns 1-17, number of conflicts (01-28) in columns 59-60, card type number 27 in columns 79-80.
COMBINE Format:	COMBINE in columns 1-7, number of combinations (01-10) in columns 59-60, card type number 28 in columns 79-80.

3. The following cards are to be changed periodically. They have no effect on the operations of the scheduling program.

(DATE CARD Format):	The date the schedule will be actually used may be keypunched beginning in column ten. This card is to be inserted as a separate data set immediately following the course request deck.

(Message cards):	Provision has been made for the insertion of message cards in the same data set immediately following the DATE card. Two message cards are available for each class, and two message cards for teachers.

Distribution of control cards is dependent upon the system of operation selected by the school. In a central operation, teachers would submit requests on an appropriate form. Modifications of this would place decks of cards in the hands of individual teachers in sufficient quantity to permit them to submit request decks in various stages of completion.

Many schools might wish to issue primary control cards (COURSE, PERIODS, STUDENT, STUDENT ALL, and STUDENT NONE) to staff members. Others may wish to also issue one or more decks of optional control cards (TEACHER, ROOM, GROUP, MAY, OPEN, OPEN MAY, CONFLICT MAXIMUM, etc.). It is recommended that the updating cards (ADD, CHANGE, DELETE, PURGE) as well as certain specialized (and conflict-intensifying) cards such as STARTING, BETWEEN PERIOD, AND PERIOD, and PRIORITY remain in the hands of the coordinator, to be issued as circumstances warrant.

Preparing Course Request Decks for the DDCS System

In Daily Demand Computer Scheduling, teachers submit requests for students to be scheduled. They may request any or all of the students enrolled for a particular course, and several requests may be submitted for each course. Each request, besides specifying which students are wanted, also indicates how much time is desired. Further, a teacher may specify precisely when and where he wants the group to meet; but normally he will let the computer select a time and place so that fewer conflicts will be created with other teachers' requests. The requests are assembled and taken to the computer for processing. Final print-out provides individual student schedules, teacher schedules, corresponding class rolls, and administration reports. Generally, teachers have control over: (1) length of class time, (2) frequency of class contact, (3) grouping, and (4) mode of instruction: open individualized laboratory vs. teacher-directed group activities.

Submitting Scheduling Requests

All control cards in the Daily Demand Computer Scheduling System (DDCS) should be prekeypunched in sufficient quantities to provide personnel with the necessary cards to make requests appropriate to their own academic or administrative area.

Basic Control Cards

Every request submitted to the computer must contain the following basic control cards regardless of other cards (optional control cards) which may also be included in the request deck:

1. One COURSE card
2. One PERIODS card

3. One or more STUDENT cards or one STUDENTS ALL card or one STUDENT NONE card.

The COURSE card. This card determines for which course the request is being made. It is the only card for which the sequential order in the deck is critical--that is, it must always appear at the beginning of each new request. All other cards may appear at any point in the request deck.

The PERIODS card. The PERIODS card is used to specify the number of periods for which a given request is being made. A school may select the length of time for each period.

The STUDENT card. The STUDENT card designates which student or students are to be scheduled for the class. At least one STUDENT card must be included in each request deck, except that a STUDENTS ALL may be substituted to call all of the students currently enrolled in a given course, or a STUDENTS NONE may be substituted to prevent a teacher and/or room from being scheduled with any students for a specified time.

The STUDENT ALL card. The STUDENTS ALL card is a convenient device which permits a teacher to request all of the students enrolled in a given course without having to include STUDENT cards for each enrollee.

The STUDENTS NONE card. The STUDENTS NONE card is used to block a teacher and/or room from being scheduled for instructional activities for the period of time designated by the PERIODS card which has to be included in the same request deck.

Optional Control Cards

The optional control cards are not required in a request deck unless the person making the request wishes to alter it in some way from the most basic type of request.

The TEACHER card. The TEACHER card should be inserted in the request deck when the user wishes to designate some teacher or teachers other than those recorded in the master file as the instructor for students involved in the request. Any insertion of one or more TEACHER cards automatically overrides all teachers who have been orginally assigned for that course for the master file.

The TEACHER NONE card. The TEACHER NONE card is used when it is elected not to designate a particular teacher for the course.

The ROOM card. The ROOM card should be inserted in the request deck when the user wishes to designate some room other than those listed in the master file as appropriate for that particular course.

The ROOM NONE card. The ROOM NONE card is used when no special room assignment is necessary.

The CONFLICTS MAXIMUM card. The program carries an automatic CONFLICTS MAXIMUM of one-third of the class, which is computed individually for each request based on the number of students involved. However, it is possible to specify the maximum number of conflicts that can be tolerated for a given course request by using a CONFLICTS MAXIMUM card.

The NO SHORT card. Wherever possible the computer schedules for the exact number of periods specified by the teacher. When the number of students in conflict reaches beyond the CONFLICTS MAXIMUM, the number of periods will be cut down to determine if the request can be scheduled within the CONFLICTS MAXIMUM boundaries for less time.

However, if the teacher would prefer to have the request bumped rather than have the amount of time cut down, the NO SHORT card would be inserted into the request deck.

The NO INDIVIDUAL SHORT card. The NO INDIVIDUAL SHORT card is a restrictive card that is used to prevent students from being scheduled to enter late or to leave early.

The NO LATE SHORT card. The NO LATE SHORT card is designed to insure that all students who are scheduled for a given class are scheduled to arrive at the given time. It is still possible for some of them to be scheduled to leave early, however.

The GROUPS card. The GROUPS card is used in a request when it is assumed that the teacher will conduct the same activity more than one time and the students are scheduled into which ever session is most convenient. When such a request is processed, the computer will divide the total number of students into the number of groups specified in the GROUPS card. In brief, (1) students are assigned into specific groups; (2) the starting time of each group is specified by the computer; (3) students are required to attend at the specified time.

The MAY SCHEDULE card. The MAY SCHEDULE card causes processing which is similar to that of the GROUPS card in that it divides the total number of students into the number of groups specified on the MAY card. However, if the MAY SCHEDULE card is used, students will not be obligated to schedule the indicated activity but rather it becomes an elective for that day.

The OPEN card. The OPEN card is used to open up a block of time during which the teacher or staff member will remain on duty to receive students for instruction, testing laboratory work, consultation, or other activities. The length of the block of time is indicated on the OPEN card. The students will stay for the time indicated by the PERIODS card of the same request. In brief, the student must attend for a specified number of periods, sometimes during the OPEN block of time.

The OPEN (MAY) card. The OPEN (MAY) card causes processing which is somewhat similar to that of the OPEN card in that it specifies a block of open time. The difference is that if the OPEN (MAY) card is used, most students will not be obligated to schedule the indicated activity, but rather it becomes an elective for that day which they they may sign up for on their unscheduled time.

The MAY SCHEDULE and the OPEN (MAY) differ in that the OPEN (MAY) allows students to select the number of consecutive periods specified within a larger block of time while the MAY SCHEDULE restricts the students to start the activity at the first period of each group of consecutive periods. It should be noted that GROUPS, OPEN, MAY and OPEN (MAY) are mutually exclusive. That is, only one card selected from these four options may be used in a given course request.

The NO PRIORITY card. A request containing the NO PRIORITY card constitutes an alternate request which the computer may turn to in case a regular request could not be scheduled for a given teacher.

Coordinator Issued Control Cards

Frequently, special situations exist which cause the regular use of a coordinator issued control card. The facility for responding to particular requirements is an inherent part of the DDCS system, but any request that tends to reduce the flexibility of the system should be avoided if the scheduling can be accomplished in some other way.

The STARTING card. The STARTING card specifies that the request must commence at a specified time. PRIORITY processing is usually also requested in connection with a request involving a starting time.

The BETWEEN PERIOD and AND PERIOD (INCLUSIVE) cards. A request containing a combination of BETWEEN PERIOD - AND PERIOD (INCLUSIVE) cards indicates that the request must start and end within the limits of time specified by the cards.

The PRIORITY card. A PRIORITY card virtually assures that the request will be scheduled without conflict. This is possible because with the addition of a PRIORITY card or cards to a request deck, the request is moved to the front of the scheduling where there are no conflicts or limitations placed on the computer's selection of a proper time to schedule the request.

360/50 Daily Demand Computer Scheduling Program

A print-out of the complete 360/50 Daily Demand Computer Scheduling Program is shown in Part VI, Appendix B. The complete program was placed in the Appendix due to the technical nature of the program.

General Guidelines

It might be appropriate to establish a procedure in which the frequency of pupil-teacher contacts is reviewed by the teacher in order that no pupil is neglected in his progress.

Record-keeping is extremely important in any continuous progress program. The teacher must be constantly alert to determine the optimum amount of time in each learning phase for each individual student. In addition, it is necessary to continually update course requests as a result of conflicts that may have occurred. The operation of the scheduling program must be visualized as a continuous sequence requiring constant modification--indeed, it can be considered parallel to learning in that respect.

Operational Routines

It is recommended that a series of operational routines be established. These include the following procedures:

1. Determination of daily program of instruction and learning.
2. Preparation of course request decks, including handling of special requests and coordinator-issued cards.
3. Modifications of previous days' schedules.
4. Insertion of special features such as DATE and MESSAGE cards.
5. The scheduling function, including deadlines and time in transit.
6. Distributing output data and individual schedules to staff.
7. Analyzing for major discrepancies.
8. Making appropriate corrections and modifications.
9. Distributing schedules to students.
10. Completing student schedules, including lunch and use of unscheduled time.
11. Utilizing statistical routines to help maintain scheduling equilibruim.

It is recognized that individual schools will have individual requirements. The program consultants can be of service in helping to analyze the existing situation and making recommendations for modification.

Evaluation

Provisions should be made for continuous evaluation. The evaluations should be objective as well as subjective and should include some pertinent statistical comparisons of relative amount of time spent in each class, and the relative learning achievement. Attitudinal surveys should be compared and analyzed. These could perhaps include staff, students and/or community.

Preparations for evaluation should be made well in advance of actually beginning the scheduling project. In-service programs should be designed to more effectively involve staff in the process of evaluation.

Individual aspects of the scheduling program should be examined in detail. Are the various controls operating at optimum effectiveness? What modifications should be made to develop a more effective scheduling program? This type of program will probably never by completely developed. It should develop and change as the process of education changes.

B. Validation of the DDCS - 360/50 Program

The revised Daily Demand Computer Scheduling Program was validated using data which had previously been collected and compiled at the Brigham Young University Laboratory School during the forty consecutive days of operation of the 7040 program during 1966. This stored input request data was submitted to the 360/50 scheduling program and the resultant output was analyzed. The initial phase of the analysis consisted of the debugging process in which the preliminary output information was used to modify and improve the scheduling program itself. As has been previously mentioned, the process of modification and improvement is a continuous one. This would be true during actual operation within a school system as well as during the development stages.

Once the basic scheduling system is functioning, it is possible to obtain completed student schedules during almost any phase of development. The process of validation included a series of modifications and rescheduling using the same stored data. Analyses were made in each of the final stages and conclusions formulated.

The output data from the 360/50 scheduling process provided (1) a scheduling sequence that was significantly faster than the 7040 system; (2) student schedules for all 500 students in the initial data set (see sample in Figure 2); (3) teacher schedules for all teachers in the initial data set (see sample in Figure 3); and (4) class lists for all course requests (see sample in Figure 4).

Analysis of the scheduling efficiency, in terms of student schedule density indicated that the IBM 360/50 program had not achieved the optimum level that had been obtained in the original IBM 7040 program. A number of factors were found to account for this apparent discrepancy. First, a number of additions were made to the original program to provide more latitude in teacher requests. Additional computer routines were required to make these modifications of the instructional program possible. Second, some discrepancies in card type numbers were discovered. For example, student number 999 was the card type number for the STUDENTS ALL card in the IBM 7040 system. The IBM 360/50 system used a four-digit code (9999) in which the number 999 was simply the student number of one individual student. Consequently, requests for all students in a given course simply assigned one specific student rather than all students. Third, a need was recognized for the reassignment of priorities to various types of course requests. This modification would give singleton courses (those that are to be taught only once in a day) a higher priority than group requests, which would increase the scheduling efficiency. Fourth, the scheduling function was making only a single pass in assigning a time pattern for a course request. The need for multiple passes was readily recognized. Multiple passes were part of the 7040 computer program.

At the time of this report, the modifications mentioned in the preceding paragraphs have been programmed. The opportunity has not yet been available to test them. With these modifications, and others that may be necessary, it is anticipated that the scheduling density of the IBM 360/50 program will approach and surpass that of the IBM 7040 program in the near future.

```
MCCLEAN, KEITH         STUDENT SCHEDULE  WED APR 12, 1967                                      GRADE

    TIME     COURSE   ROOM      TIME     COURSE   ROOM      TIME     COURSE   ROOM      TIME     COURSE   ROOM

 1  8 30                       8 10 15   ENG7     280E    15 12 00                     22  1 45
 2  8 45                       9 10 30                    16 12 15                     23  2 00
 3  9 00     ISMA     344E    10 10 45                    17 12 30   SPAN     305A     24  2 15
 4  9 15     ISMA     344E    11 11 00                    18 12 45   SPAN     305 A    25  2 30
 5  9 30     ISMA     250A    12 11 15                    19  1 00   SPAN     305A     26  2 45
 6  9 45     ISMA     NOROOM  13 11 30                    20  1 15                     27  3 00
 7 10 00     ENG7     280E    14 11 45                    21  1 30                     28  3 15

                         YOU MAY CHOOSE FROM THE FOLLOWING CLASSES

 1  1 15     CLERK    240C     1  1 30   CLERK    240C     1  1 45   CLERK    240C      1  2 00   CLERK    240C
 1  2 15     CLERK    240C     1  2 30   CLERK    240C     1  2 45   CLERK    240C      1  3 00   CLERK    240C

                         MESSAGE CARD 1
MESSAGE CARD  2
```

FIGURE 2 - STUDENT SCHEDULE
B Y HIGH
IBM 360/50

```
PECK, D.                     WED APR 12, 1967   TEACHER/ADMINSTRATOR SCHEDULE

   TIME    COURSE   ROOM                    TIME    COURSE   ROOM          TIME    COURSE    ROOM

 1  8  30  MATH8    380E                 11 11 00  ALG2     380E        21  1 30
 2  8  45   "        "                   12 11 15   "        "          22  1 45
 3  9  00  MATH8    380E                 13 11 30  MATH7    380E        23  2 00
 4  9  15   "        "                   14 11 45   "        "          24  2 15
 5  9  45  ALG1     380E                 15 12 00  MATH7    380E        25  2 30
 6  9  45   "        "                   16 12 15   "        "          26  2 45
 7 10  00  ALG2      "                   17 12 30                       27  3 00
 8 10  15   "        "                   18 12 45                       28  3 15
 9 10  30  ALG2     380E                 19  1 00
10 10  45   "        "                   20  1 15
```

FIGURE 3 - TEACHER SCHEDULE
B Y HIGH
IBM 360/50

```
CLASS ROLL COURSE GEOGRA              WED APR 12, 1967          TIME 11 00 - 11 55     ROOM - 140A

MCCONKIE, D.

ALDER, LYNN                           BENNION, JOSEPH                            CHRISTENSEN, DONALD
CLARK, ROBERT                         DIXON, KENNETH                             GREEN, BRUCE
HUISH, LEE                            IVIE, SHANNON                              LAMBERT, JANINE
MURDOCK, ALAN             UNAVAILABLE NICHOLES, EDWARD                           NICHOLES, RICHARD
MANAMARA, JAY                         OAKES, CYNTHIA            11 30 - 11 55    RICHAN, DOROTHY
STEWART, DENNIS K                     TEICHERT, FRED            11 15 - 11 55    THOMAS, CORNEL
YOUNG, WAYNE                          HOWELL, PEGGY                              JOHNSON, ROBERT LEE
```

FIGURE 4 - CLASS LIST
B Y HIGH
IBM 360/50

C. Research and Testing of the 360/50 Program at a Selected Pilot School

The third terminal objective of this study was to research and test the completed 360/50 computer program with data from a sample pilot school. Valley High School at Las Vegas, Nevada was selected as the pilot school. Valley High School has 2,200 students and 94 teachers and was being scheduled by the Stanford School Scheduling System.

A research director, a system analyst, and an educational design consultant went to Valley High School to prepare the following basic master file data: schedule of rooms, teacher files, course files and student files. The key punching of all file data cards was completed for us by the Clark County School District Computer Center. Basic requests for five separate days of school operation was then obtained from the teachers and administrators at Valley High School. The requests were edited by our scheduling coordinator and then run on the 360/50 computer scheduling program. De-bugging of the computer program was again included at the beginning operations. As indicated previously, the de-bugging is a continuous process.

The output data provided 2,200 separate individual schedules, teacher schedules for all teachers and administrators, and class lists for all teachers showing the names of students who were scheduled as well as the names of students who were not scheduled on each request. Samples of Las Vegas output data are shown in Figures 5-7: a student schedule is shown in Figure 5; a teacher schedule is shown in Figure 6; and a class list is shown in Figure 7.

The output data was analyzed for completeness and efficiency with the following results:

1. All 2,200 Valley High School students were scheduled on the 360/50 computer program.
2. All teachers received a teacher schedule showing their requests which were scheduled and the time the requests were scheduled.
3. All teachers received a class list for each request showing whether the total requests could be scheduled and, if so, the names of students who were scheduled and the names of students who were not scheduled, if any.
4. The efficiency of scheduling was not as high as desired by the program authors. Therefore, an analysis of the efficiency of the program revealed the following problems:
    a. The same computer problems that were indicated in the previous discussion of validation utilizing the original Brigham Young Laboratory School data were also present with the pilot school scheduling.

```
KLEIN          ROBIN       J    STUDENT SCHEDULE  MAY 15 WED.  69
                                                                                                  GRADE

   TIME         COURSE     ROOM       TIME     COURSE      ROOM     TIME     COURSE  ROOM      TIME     COURSE    ROOM

1  8 30        ENGLISH    B-GYM     8 10 15                        15 12 00                   22  1 45
2  8 45        ENGLISH    NO ROOM   9 10 30                        16 12 15                   23  2 00  AUTO I    NO ROOM
3  9 00        ENGLISH    NO ROOM  10 10 45                        17 12 30                   24  2 15  AUTO I    NO ROOM
4  9 15        ENGLISH    NO ROOM  11 11 00                        18 12 45                   25  2 30  AUTO I    400
5  9 30                            12 11 15                        19  1 00                   26  2 45  AUTO I    NO ROOM
6  9 45                            13 11 30                        20  1 15                   27  3 00  AUTO I    NO ROOM
7 10 00                            14 11 45                        21  1 30                   28  3 15  AUTO I    NO ROOM

                         YOU MAY CHOOSE FROM THE FOLLOWING CLASSES

1  9 30        PSYCHOLO  235        1 11 15  PSYCHOLO     235     1 11 15  PSYCHOLO 235     1 11 15  PSYCHOLO 235
1 11 15        PSYCHOLO  235        1 11 15  PSYCHOLO     235     1 11 15  PSYCHOLO 235     1 11 15  PSYCHOLO 235
1 11 15        PSYCHOLO  235        1 11 15  PSYCHOLO     235     1 11 15  PSYCHOLO 235
                         MESSAGE CARD 1
MESSAGE CARD 2
```

FIGURE 5 - STUDENT SCHEDULE
LAS VEGAS - VALLEY HIGH

```
BELLAMY           MAY 15 WED. 69        TEACHER/ADMINSTRATOR SCHEDULE

     TIME     COURSE      ROOM                            TIME     COURSE     ROOM                 TIME     COURSE     ROOM

                                                       11 11 00    HUM RELA    212    OPEN REQUE 21  1 30
   1  8 30                                             12 11 15    HUM RELA    212    OPEN REQUE 22  1 45
   2  8 45                                             13 11 30    HUM RELA    212    OPEN REQUE 23  2 00
   3  9 00                                             14 11 45    HUM RELA    212    OPEN REQUE 24  2 15
   4  9 15   HUM RELA     212         OPEN REQUE       15 12 00    HUM RELA    212    OPEN REQUE 25  2 30
   5  9 30   HUM RELA     212         OPEN REQUE       16 12 15    HUM RELA    212    OPEN REQUE 26  2 45
   6  9 45   HUM RELA     212         OPEN REQUE       17 12 30    HUM RELA    212    OPEN REQUE 27  3 00
   7 10 00   HUM RELA     212         OPEN REQUE       18 12 45       "         "                28  3 00
   8 10 15   HUM RELA     212         OPEN REQUE       19  1 00       "         "
   9 10 30   HUM RELA     212         OPEN REQUE       20  1 51
  10 10 45   HUM RELA     212         OPEN REQUE
```

FIGURE 6 - TEACHER SCHEDULE
LAS VEGAS - VALLEY HIGH

```
CLASS ROLL     COURSE ENGLISH              MAY 15 WED. 69          TIME  2 30 - 3 25  ROOM 242            GROUP REQUEST

SOULE

BARNETT    SHERRY M  BELL  JR     PHILIP   E  BLACKMER   CHIRISTIE L  BONNY      VICKI    D  BUEHLER    RONALD    W
BURDETTE   MIKE   T  CAPRI        RONALD      CARTER     JAMES     M  COBURN     CALIN    R  DELELLIS   STEPHANIE J
DEVITTE    KEVIN  J  DUBUG        ALLIN    J  ENDY       TOMMY     D  FERNANDEZ  VICKI    M  FLICK      JILL      W
GORDON     CAROL  L  HARRIS       MARY     A  HAUGER     GARY      L  HAYCOCK    JULIE    C  HEINDEL    ALICE
HOPKINSON  VERNON L  JABBORA      JAKE     L  JOSE       RAMON     J  KACHULIS   GLENN    A  MCCREA     SUSAN     E
MEGYESI    PAM       MILLER       DAVID       OREILLY    BILL      G  PHILLIPS   DOUG     M  RICHARD    DEBRA     A
SCHWAB     LOIS   A  SMITH        CAROLYN     WANLESS    KAREN     S  WILLIS     ALETHIA  -  WRANGHAM   CYNTHIA
```

FIGURE 7 - CLASS LIST
LAS VEGAS - VALLEY HIGH

b. The course structures at the pilot school were not designed with the Daily Demand Computer Scheduling System in mind. As a result, some of the course request groupings had to be somewhat artificial to compensate for the nature of the data in the master file.

It is expected that the items indicated in (a), above, will have been solved by the modifications that have already been programmed--as indicated in the previous section. The problem mentioned in (b) is unique to the pilot school. It can only be solved by designing course structures, as well as the entire educational system, for the Daily Demand Computer Scheduling program.

PART V. CONCLUSIONS AND RECOMMENDATIONS

A. Conclusions

B. Recommendations

PART V. CONCLUSIONS AND RECOMMENDATIONS

A. Conclusions

The preparation, testing, and validation of a computer program of this magnitude is an on-going process. While this study made significant progress toward a Daily Demand Computer Schedule which can be used with all sizes of schools and under varying circumstances, it is recognized that the optimum schedule has not been achieved. The limitations of time, as well as equipment and personnel availability, require additional testing prior to achieving the desired level of efficiency. From the progress made to date, a number of significant conclusions can be made.

1. The Daily Demand Computer system is a viable tool that has real implications in extending the scope of individualized instruction.
2. Daily Demand Computer Scheduling can be successfully accomplished in a small high school using the program designed for the IBM 7040 computer.
3. The IBM 360/50 Daily Demand Computer Scheduling system is basically complete, and can handle the expanded parameters required for the scheduling of a large comprehensive secondary school.
4. Additional modifications are still being tested to increase the efficiency of the 360/50 scheduling program.
5. The educational modifications of the IBM 360/50 program have expanded the ability of the teacher to individualize instruction through the management of time.
6. The IBM 360/50 schedule provides time for student independent study and promotes certain student discretionary activities.
7. The variable time and grouping patterns of the Daily Demand Computer Schedule encourage planning for specific types of unique instructional activities.
8. The Daily Demand Computer Schedule encourages the teacher to build schedules involving specific activities for individual students.
9. The Daily Demand Computer Schedule permits the structuring of permanent and/or temporary groups. The grouping can be varied on any given day or based on the teacher-established instructional parameters.
10. The need for additional testing and evaluation of the program in actual operation is desirable in order to increase efficiency and reduce conflicts.
11. Any schedule that has an individualized instruction--learning program--and has access to an IBM 360/50 computer can now move to a Daily Demand Computer Schedule.

B. Recommendations

Based upon the preceding findings and conclusions, the following recommendations are made:

1. It is recommended that any small school that has access to the IBM 7040 computer or any school that has an IBM 360/50 computer available, that is concurrently involved in an individualized continuous progress educational program, consider the possibility of utilizing the Daily Demand Computer Schedule.
2. It is recommended that modification and development of the Daily Demand Computer Schedule be continued until optimum scheduling efficiency is obtained.
3. It is recommended that the process of modification and evaluation of the Daily Demand Computer Schedule be directed toward the continued emphasis of individualization of instruction.
4. It is recommended that the Daily Demand Computer Schedule be placed in a functional, continuous operation at some pilot school at the earliest possible date.
5. It is recommended that extensive staff and student orientation be provided prior to the assumption of the Daily Demand Computer Schedule.

PART VI. SUPPLEMENTARY AND APPENDIX MATERIALS

A. References

B. Bibliography

C. Glossary

D. Supplementary and Appendix Materials

A. References

Footnote references have been listed on the bottom of each page where the footnote occurs.

B. Bibliography

Allen, Dwight W., and Donald Delay. "Stanford's Computer System Gives Scheduling Freedom to 26 Districts," The Nation's Schools, 77:121-5, March, 1966.

Anderson, Edward J. "Wayland High School's Flexible Scheduling," Journal of Secondary Education, 36:354-6, October, 1961.

Anderson, G. Ernest, Jr. "New Computer Programs to Help Flexible Schedules," The Nation's Schools, 81:67, June, 1968.

Arnett, M. Rex. Daily Demand Computer Scheduling: Questions and Answers From a Student's Viewpoint. Provo, Utah: Education Experimental Programs, 1967.

Arnett, M. Rex. "A Manuscript on Daily Demand Computer Scheduling at B.Y.U Laboratory School," Unpublished manuscript, Stanford University, 1967.

Arnett, M. Rex. "A Report to the Parent-Teachers Association on the Daily Demand Computer Scheduling Project." Provo, Utah: Brigham Young University Laboratory School, 1967.

Beggs, David W., III. Decatur-Lakeview High School. Englewood Cliffs, New Jersey: Prentice-Hall, 1964.

Braddock, Clayton. "Changing Times Are Changing Schools," Southern Education Report, 3:24-30, October, 1967.

Brown, Charles E. "Towards Quality Education at Newton," Journal of Secondary Education, 36:357-60, October, 1961.

Bush, Robert N. "A Searching Appraisal of New Developments," Journal of Secondary Education, 37:321-6, October, 1962.

Bush, Robert N., and Dwight W. Allen. "Flexible Scheduling for What?" Journal of Secondary Education, 36:346-53, October, 1961.

Bush, Robert N., and Dwight W. Allen. A New Design for High School Education. New York: McGraw-Hill, 1964.

Bushnell, Don D., and Dwight W. Allen. The Computer in American Education. New York: John Wiley and Sons, 1967.

Byhre, Edward, B. "An Enabling Act: Bloomington's Flexible Modular Scheduling," Minnesota Journal of Education, 49:11-13, September, 1968.

D'Antuono, Anthony, and William Shaw McCallum, Jr. "Constructing the Master Schedule by a Computer," National Association of Secondary School Principals Buttetin, 49:58-65, October, 1965.

Davis, E. Dale. Focus on Secondary Education. Glenview, Illinois: Scott Foresman and Company, 1966.

Duncan, John Richard. "A Study of Teachers' Expectations in Three Selected Secondary Schools." Unpublished doctor's dissertation, Indiana University, 1967.

Flanagan, Scott. "Machine Programming at Huntington Beach High School," Journal of Secondary Education, 36:371-3, October, 1961

Flowers, John F. "The Production of Secondary School Timetables by Digital Computer," Educational Data Processing Newsletter, 4:6-9, January, 1965.

Georgiades, William, and Joan Bjelke. "Evaluation of English Achievement in a Ninth Grade, Three-Period Team-Teaching Class," California Journal of Educational Research, 17:100-112, May, 1966.

Goodlad, John I., and Robert H. Anderson. The Nongraded Elementary School. New York: Harcourt, Brace & World, 1963.

Greenberger, Martin (ed). Computers and the World of the Future. Cambridge, Massachusetts: The M.I.T. Press, 1962.

Grossman, Alvin, and Robert L. Howe. Data Processing for Educators. Chicago: Educational Methods, Inc., 1965.

Gruman, Allen J. "A Comparison of Computer Programs for the Assignment of Students to Calsses: Report of an Experimental Study," Journal of Educational Data Processing, 4:7-15, Winter, 1966-67.

Hamilton, Jack A., and Robert F. Madgic. "Can Flexible Scheduling Improve Social Studies Instruction?" Journal of Secondary Education, 39:295-298. November, 1964.

Heathers, Glen. "School Organization: Nongrading, Dual Progress, and Team Teaching," The Changing American School, sixty-fifth yearbook of the National Society for the Study of Education, Part II. Chicago: University of Chicago Press, 1966.

Harmon, Johnson W. "Unlocking Schoo-Shop Doors via Flexible Scheduling," School Shop, 28:40-42, November, 1968.

Hoffman, Orrin. "Flexible Schedule," Journal of Secondary Education, 43:278-282, October, 1968.

Kaimann, Richard A., and Robert W. Marker (ed.). Educational Data Processing: New Dimensions and Prospects. Boston: Houghton Mifflin Co., 1967.

Kapfer, Philip G. "An Instructional Management Strategy for Individualized Learning," Phi Delta Kappan, 49:260-263, January, 1968.

Laurits, James. "Approaches to Flexibility at Cubberley Senior High School," Journal of Secondary Education, 36:361-363, October, 1961.

Llana, Andres, Jr. "A Computer Based Student Scheduling System," Educational Data Processing Newsletter, 4:11-14, January, 1965.

Lobb, M. Delbert. "A Basis for First Steps in Flexible Scheduling." Journal of Secondary Education, 36:367-370, October, 1961.

Loughary, John W. Man-Machine Systems in Education. New York: Harper and Row, 1966.

Manlove, Donald C., and David W. Beggs, III. Flexible Scheduling. Bloomington, Indiana: Indiana University Press, 1965.

Morphet, Edgar L., and David L. Jesser (ed.). Planning for Effective Utilization of Technology in Education. Report of an Eight-State Project. Denver, Colorado: Bradford Printing, August, 1968.

Murphy, Judith. School Scheduling by Computer--The Story of GASP. New York: Educational Facilities Laboratories, 1964.

National Society for the Study of Education. The Changing American School. Sixty-fifth Yearbook, Part II. Chicago: University of Chicago Press, 1966.

Oakford, Robert V. "Machine Assistance for Constructing the High School Schedule: An Industrial Engineer's Report," Journal of Secondary Education, 36:374-379, October, 1961.

Oakford, Robert V., Dwight W. Allen, and Lynne A. Chatterton. "School Scheduling--Practice and Theory," Journal of Educational Data Processing, 4:16-50, Winter, 1966-67.

Ovard, Glen F. (ed). Change and Secondary School Administration. New York: The MacMillan Company, 1968.

Ovard, Glen F. A Model for Developing an Individualized, Continuous Progress Curriculum Unit Emphasizing Concepts and Behavioral Objectives. Provo, Utah: Education Experimental Programs, 1967.

Ovard, Glen F. Overview of Eight Years of Development in Continuous Progress Education. Provo, Utah: Education Experimental Programs, 1967.

Ovard, Glen F. (ed). Summary: Background, Basic Premises and Objectives of Continuous Project Programs. Provo, Utah: Education Experimental Programs, 1966.

Ovard, Glen F. and others. Description, Systems Summary and Source Program of Daily Demand Computer Scheduling. Provo, Utah: Education Experimental Programs, 1967.

Ovard, Glen F., and others. Operational Procedure, Program Control Manuals, Program Flow Charts for Daily Demand Computer Scheduling. Provo, Utah: Education Experimental Programs, 1967.

Pack, Charles Randall. "Mechanization of a Variable High School Schedule." Unpublished engineer's thesis, Stanford University, 1961.

Richardson, Duane. "Pupil Scheduling by Computer," Educational Data Processing: New Dimensions and Prospects. Boston, Massachuetts: Houghton Mifflin Company, 1967.

Shanus, Rosemary, and others. "Modular Scheduling: How Much Fact--How Much Illusion?" Business Education World, 49:9-10, 26, October, 1968.

Shelton, Landon O. "Scheduling the IBM Way," Clearing House, 39:521-525, May, 1965.

Speckhard, Gerald Paul. "An Evaluation of the Educational Program of A High School Using a Modular Schedule." Unpublished doctors, dissertation, University of Colorado, Boulder, 1966.

Storlie, Theodore R. "Evaluating Flexible Scheduling," Educational Leadership, 25:177-183, November, 1967.

Swenson, Gardner, and Donald Keys. Providing for Flexibility in Scheduling and Instruction. Englewood Cliffs, New Jersey: Prentice-Hall, 1966.

TIES Student Scheduling--User's Manual: 1969-70. Minnesota School Districts Data Processing Joint Board, 1969. (Mimeographed.)

Trump, J. Lloyd. "Developing and Evaluating a Class Schedule to Help Each Pupil Learn Better," Journal of Secondary Education, 36:338-45 October, 1961.

Trump, J. Lloyd. "Flexible Scheduling: Fad or Fundamental?" Phi Delta Kappan, 44:367-371, May, 1963.

Trump, J. Lloyd, and Delmas F. Miller. Secondary School Curriculum Improvement. Boston: Allyn and Bacon, 1968.

U. S. Department of Health, Education and Welfare. Flexibility for Vocational Education Through Computer Scheduling. Quarterly Report, April 1, 1967.

Van Dusseldorp, Ralph A., and Duane E. Richardson. "Class Scheduling by Computer," Journal of Educational Data Processing, 4:51-60, Winter, 1966-67.

Webb, L. Leon. "Selected Transitional Effects for Students Transferring from a Continuous Progress Program to a Non-Continuous Progress Program." Unpublished doctor's dissertation, Brigham Young University, 1969.

Whitlock, James W. Automatic Data Processing in Education. New York: The MacMillan Company, 1964.

Wiley, W. Deane, and Lloyd K. Bishop. The Flexibly Scheduled High School. West Nyack, New York: Parket Publishing Company, 1968.

Wulff, B. H. "Data Processing for Student Scheduling: An IBM Approach," Journal of Secondary Education, 36:380-81, October, 1961.

Yerby, Robert M. "Flexible Scheduling: Some Critical Questions and an Experiment," Journal of Secondary Education, 37:197-200, April, 1962.

C. Glossary

Continuous Progress Program - an educational program which provides for the continuous progress of students at rates and directions commensurate with their own abilities and interests, with no artificial progress barriers such as grade levels and promotion dates imposed to restrict or force students into unrealistic predetermined patterns.

Control Cards - previously prepared data processing cards that are used to provide the input data for individual course requests in the DDCS program.

Daily Demand Computer Schedule (DDCS) - computer schedule generated daily which allows teachers and students to schedule time according to the specific needs of each individual.

Generalization of Schedule - adjusting the parameters and mechanics of the computer schedule so that it might be used by any school that is involved in an individualized instructional program.

Independent Study Time - periods of unscheduled time during which the student has freedom of choice in the selection of his learning activities.

Individual Student Conflict - the overlapping of two or more sets of course requests or time reservations such that it is impossible for the given individual to satisfy the initial requests on the schedule for a given day.

Individualized Instruction - educational program designed to take into consideration individual differences in such factors as experience, methods, and rate of learning.

Instructional Management - the procedure by which the logistics of students, time, space, and instructional materials are manipulated to provide an effective learning situation.

Large Group Instruction - teacher-centered instruction that is presented to a group of students using primarily lecture methods. The differentiating feature is the instructional pattern, not the size of the group.

Learning Laboratory - area designed for the utilization of a variety of instructional materials in the facilitation of learning in a particular subject-matter area.

Nongraded Program - System of instructional management that is designed to allow placement of students at a specific instructional level in a given area of curriculum as determined by their individual needs and progress.

Small Group Instruction - characterized by participation and interaction of all group members; is frequently student-initiated and student-directed.

Teacher Request - instructional program design which includes requests for specific students, classrooms and time requirements for a given day.

Teacher Request Conflict - the overlapping of two or more sets of requests (i.e., students, rooms, time, teachers) such that it is impossible to completely satisfy the initial course requests on the schedule for a given day.

Team Teaching - cooperative efforts of two or more teachers in a specific instructional program; may also include the concept of the master teacher and/or the paraprofessional.

Traditional School Program - instructional framework where the subjects taught, time patterns, and group size remain relatively fixed and constant for at least a semester; is normally characterized by a teacher-centered program, lecture as a primary method of presentation, and little allowance for individual differences.

Validation of DDCS Schedule - comparison of revised computer schedule with earlier models--also implies that necessary revisions will be made until the new program meets accepted norms.

Variable Time Pattern - system whereby the amount of time allocated to a particular instructional activity can be varied.

D. Supplementary and Appendix Materials

Appendix A

Daily Demand Computer Scheduling Program IBM 360/50
Link Map Report

Appendix B

Daily Demand Computer Scheduling Program IBM 360/50
Print-out Computer Program

Appendix C

Daily Demand Computer Scheduling Program IBM 360/50
De-bugging Routines

Appendix A

Daily Demand Computer Scheduling Program IBM 360/50

Link Map Report

MASTER FILE BUILDER

This program reads in appropriate cards and creates the six master files on disk storage.

These files are:

1. A pointers file used to access the course information file. It also has numbers showing which students are in which grades.
2. The course information file.
3. The course name file.
4. The room name file.
5. The student name file.
6. The teacher name file.

COPYOUT

This program copies the six master files from disk onto a tape.

These six files are the only files preserved from one run to the next.

COPYIN

This program does the reverse of COPYOUT in that it copies the six master files onto a tape.

MASTER UPDATE

Programs, ADDMOD, DELMOD, CHAMOD plus their sub programs perform respectively additions, delections, and changes to the course information file and alter the pointers file accordingly.

Program NAMCON plus its sub programs performs additions, deletions, and changes to the master name files.

Note: THE MASTER UPDATE PROGRAMS DO NOT HAVE TO BE RUN WITH EVERY SCHEDULING RUN.

CONDNSE

This program prepared the student schedule file for future use. It reserves only enough space on disk storage to handle the students appearing in the student name file. It also creates a pointers file for use in accessing the student schedule file.

PREPARE

This program reads the request decks and arranges the requests by priority with highest priority first. It also fills in information needed by the scheduling programs when necessary, using the course information file to do so.

SCHEDULE

The MAIN program causes the scheduling to take place in five stages or rounds:

Round one. This round uses programs RGSKED (regular requests), GROUP (group requests), MAY (may requests), and OPENMY (open may requests) as necessary to schedule the various types of requests. Each of these programs use the sub programs indiciated on the LINK MAP to accomplish its purpose.

This round does not examine NO PRIORITY requests, nor does it attempt to allow students to individually leave the scheduled class early or arrive late. (GROUP and MAY will allow this type of scheduling).

Round two. Round two reschedules regular requests using programs RESET, DRPSKD and RGSKED. This round continues to cycle through the regular requests until a cycle shows an improvement over the previous cycle of less than 5%. No attempt is made to allow students to enter late or leave early.

Round three. Round three is a final cycle through the regular requests, allowing students to leave early and/or arrive late (subject to restrictions made by the requesting teacher). Program FILSKD is used for this type of processing.

Round four. This round is the same as round one except only the NO PRIORITY requests are processed.

Round five. Round five is the last round and is the same as round three except only the NO PRIORITY requests are processed.

The scheduling routines produce the following files:

1. The request file is altered to show how and when the requests were actually scheduled.

2. The student schedule file.

3. Programs MAY and OPENMY produce a file showing which students were requested for each of these requests.

4. The room schedule file.

MAY SORT

This is a system sort to arrange the MAY data in ascending order by student number.

STATS

This program prints out statistics on number of requests by type, requests shortened or denied, and a temporary file of similar information on students.

STAT SORT

This is a system sort on students statistics in ascending order by student number.

STUSTAT

This program prints out information by student on how many requests were made for him, how many were denied, and a breakdown of his schedule by periods scheduled in each type of request.

WRYT OUT

Prints out in the following order:

1. Room schedules (program RUMRYT).
2. Course rolls (programs CRSRYT and GRPRYT).
3. Student schedules (program STURYT and sub programs).

Programs CRSRYT and GRPRYT produce a file containing information for the printing of teacher schedules. They also produce two temporary files of room information used by STURYT in producing the student schedules.

TECH SORT

This is a system sort to arrange the teacher information from WRYTOUT preparatory to the information being printed by TECRYT.

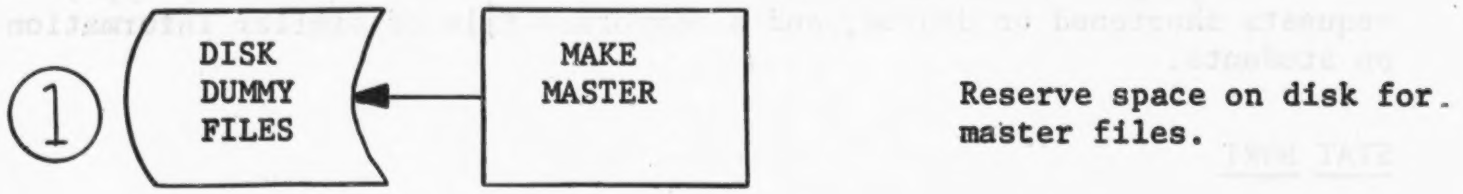
TECRYT

Prints out teacher schedules.

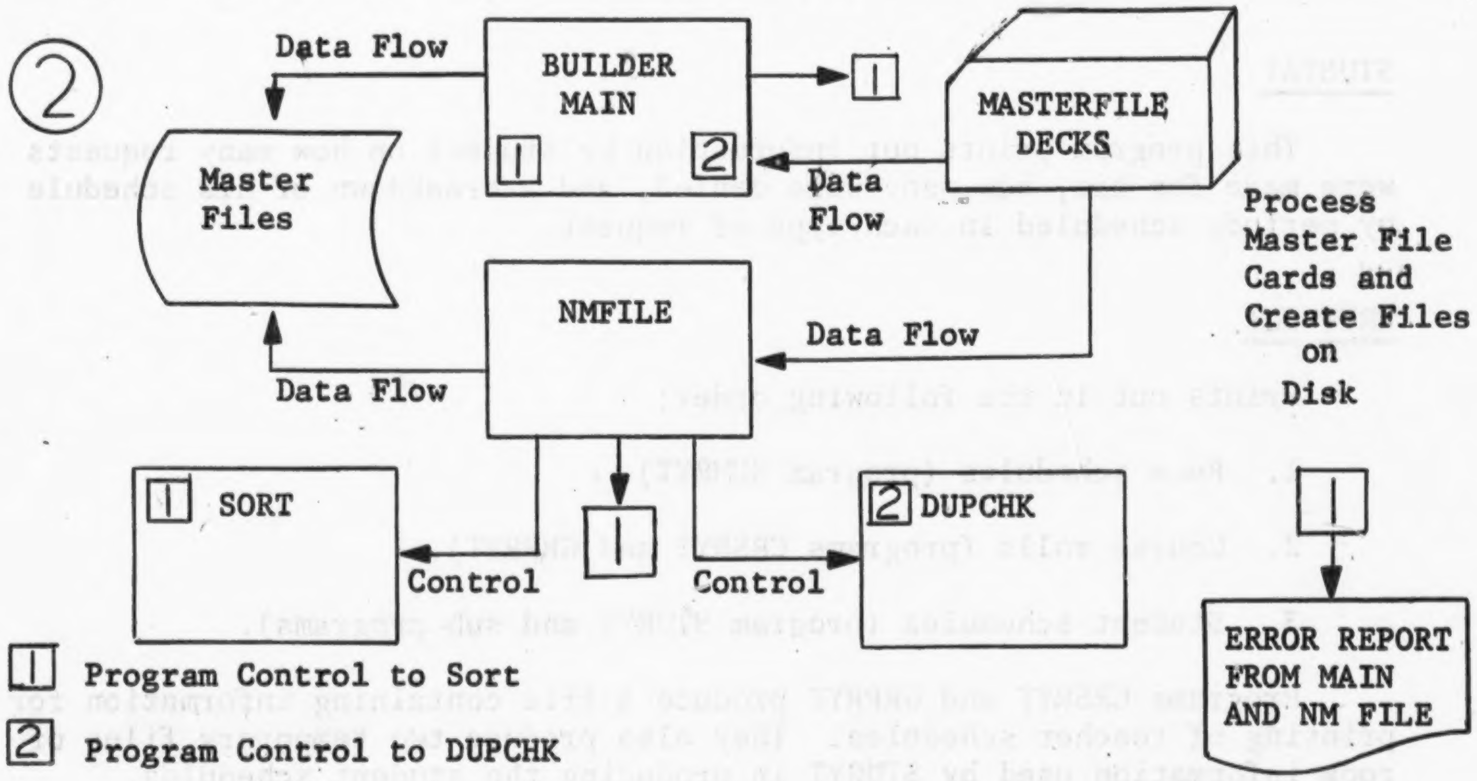
At this point COPYOUT would be executed if any alterations had been made to the master files.

DDCS SYSTEM FLOW AND LINK MAP

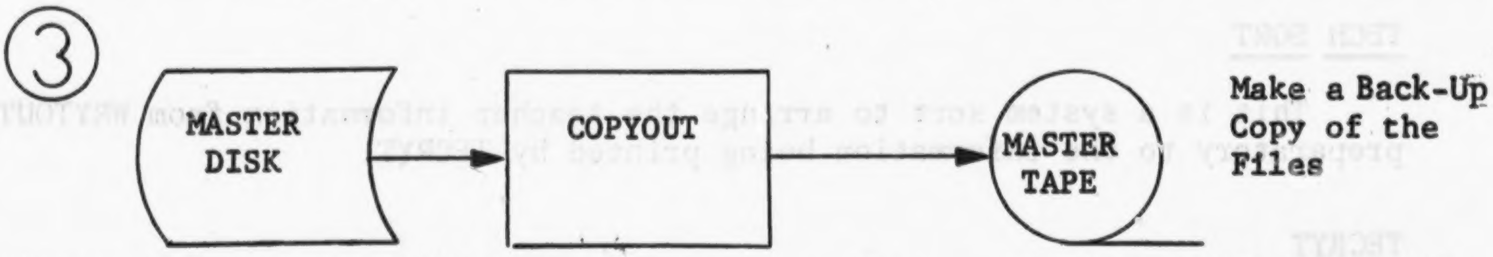
The dashed lines across the pages indicate the job-step breaks. Ⓝ N is sequence number of the step.



①
DISK DUMMY FILES
MAKE MASTER
Reserve space on disk for master files.

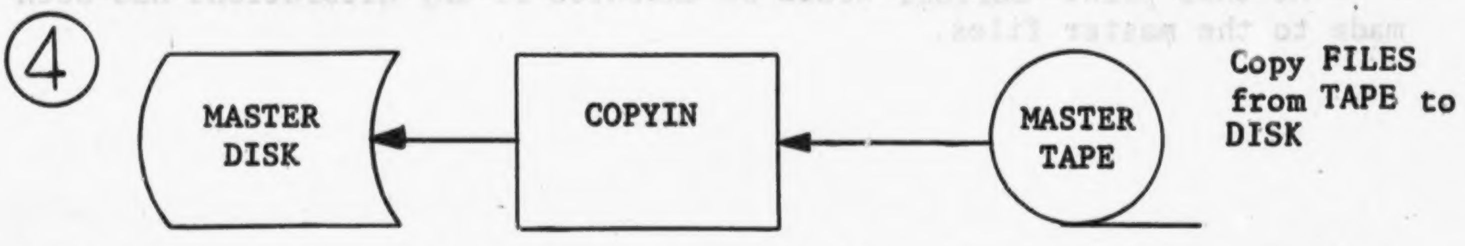


②
Data Flow
BUILDER MAIN
1
2
MASTERFILE DECKS
Master Files
Data Flow
Process Master File Cards and Create Files on Disk
NMFILE
Data Flow
Data Flow
1 SORT
1
2 DUPCHK
Control
Control
1
ERROR REPORT FROM MAIN AND NM FILE
1 Program Control to Sort
2 Program Control to DUPCHK



③
MASTER DISK
COPYOUT
MASTER TAPE
Make a Back-Up Copy of the Files

Note: This step would be run after every update on the master files as well as after the files creation.



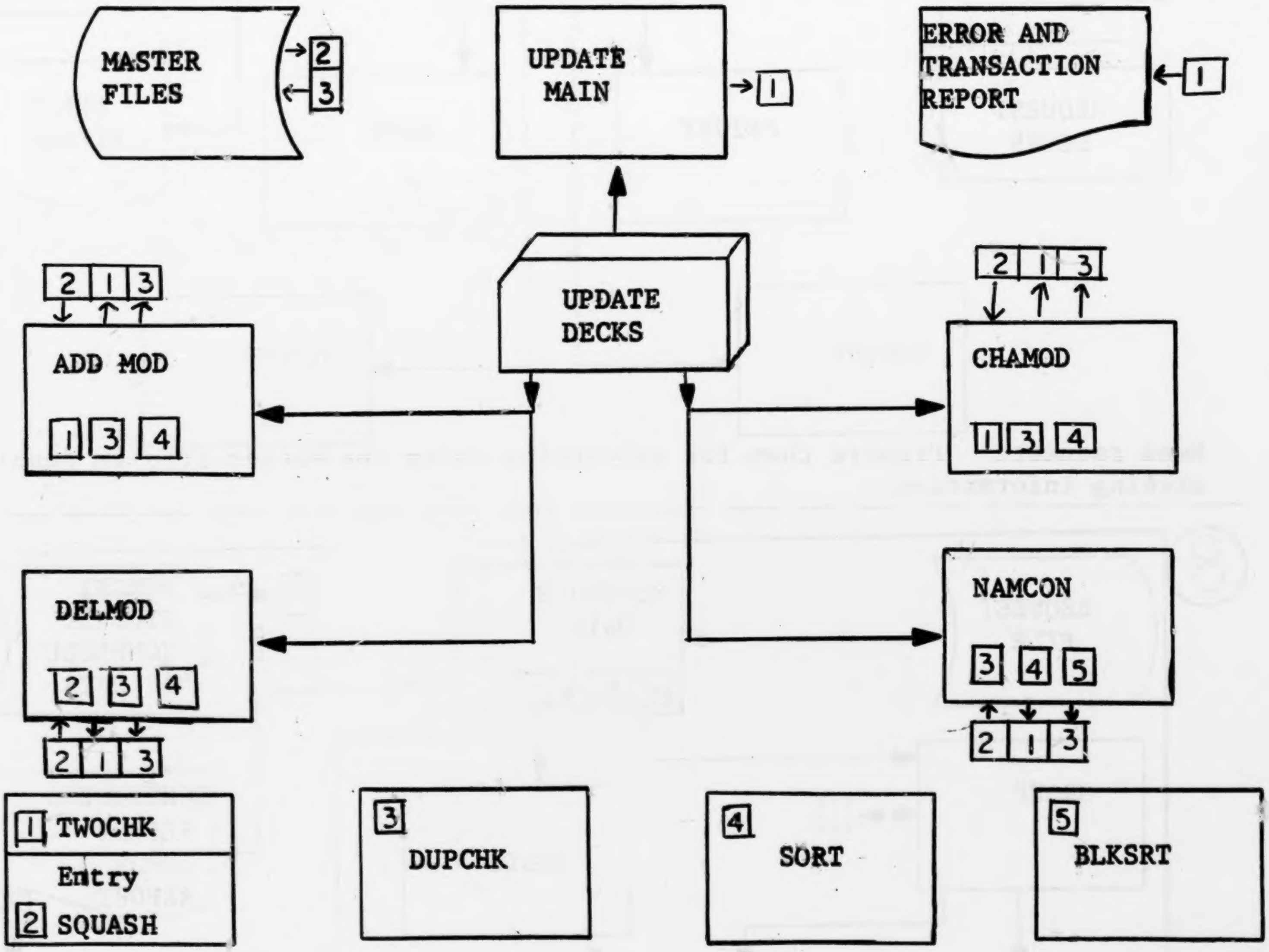
④
MASTER DISK
COPYIN
MASTER TAPE
Copy FILES from TAPE to DISK

Note: This step would only be run if the files are not resident on disk.

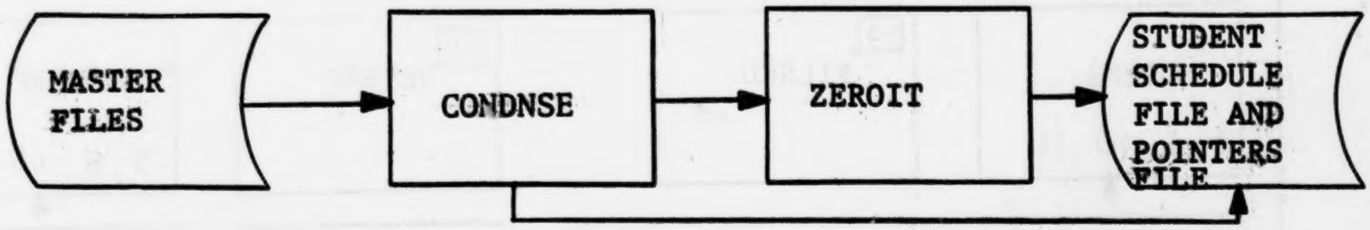
DDCS SYSTEM FLOW AND LINK MAP

NOTE: This step is run only when changes need to be made to the master files.

⑤

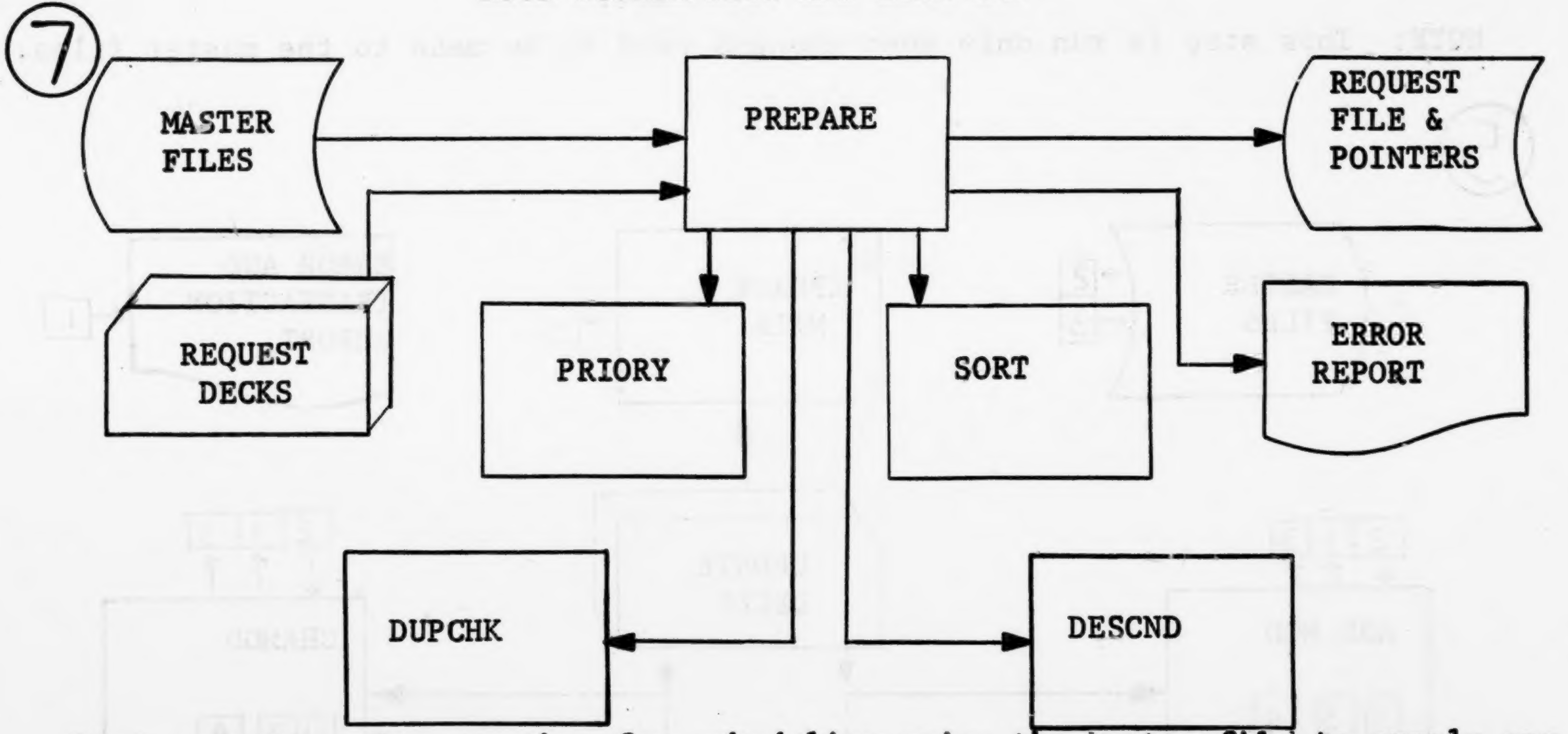

MASTER FILES
2
3
UPDATE MAIN
1
ERROR AND TRANSACTION REPORT
1
UPDATE DECKS
2 1 3
ADD MOD
1 3 4
2 1 3
CHAMOD
1 3 4
DELMOD
2 3 4
2 1 3
NAMCON
3 4 5
2 1 3
1 TWOCHK
Entry
2 SQUASH
3 DUPCHK
4 SORT
5 BLKSRT

⑥

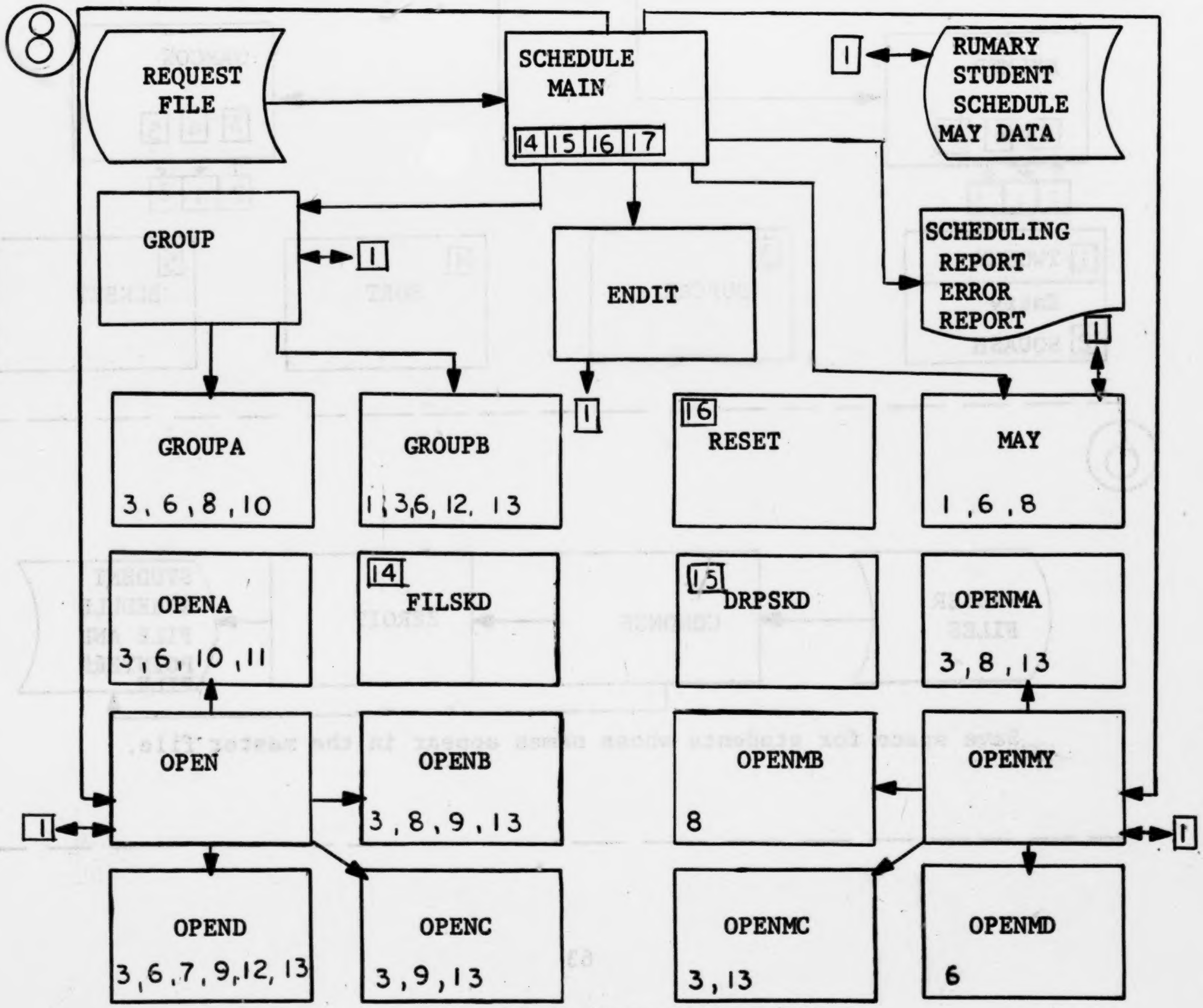

MASTER FILES
CONDNSE
ZEROIT
STUDENT SCHEDULE FILE AND POINTERS FILE

Save space for students whose names appear in the master file.

DDCS SYSTEM FLOW AND LINK MAP

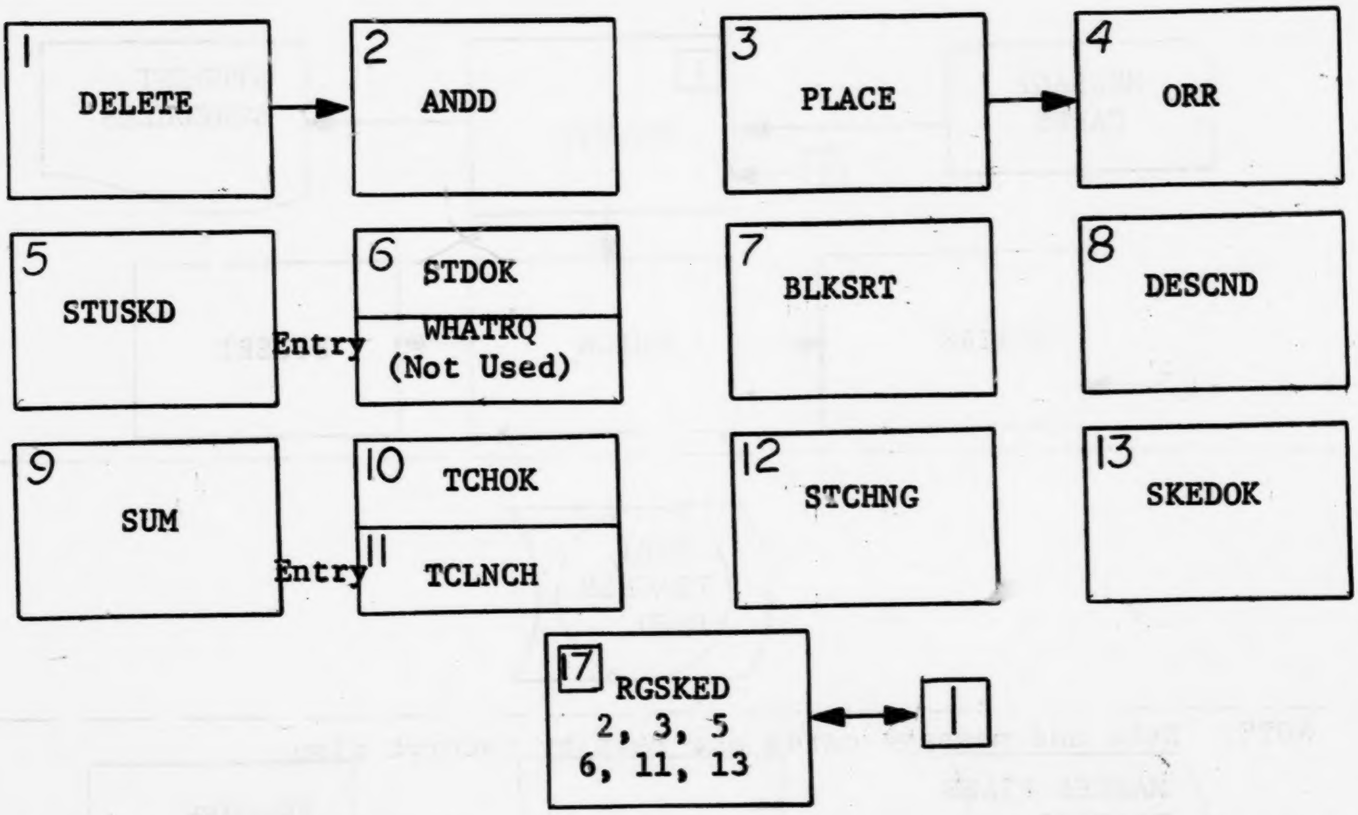

7
MASTER FILES
REQUEST DECKS
PREPARE
REQUEST FILE & POINTERS
PRIORY
SORT
ERROR REPORT
DUPCHK
DESCND

Read requests. Prepare them for scheduling using the master file to supply any missing information.


8
REQUEST FILE
SCHEDULE MAIN
14 15 16 17
1
RUMARY STUDENT SCHEDULE MAY DATA
GROUP
1
ENDIT
SCHEDULING REPORT ERROR REPORT
1
GROUPA
3, 6, 8, 10
GROUPB
1, 3, 6, 12, 13
1
16
RESET
MAY
1, 6, 8
OPENA
3, 6, 10, 11
14
FILSKD
15
DRPSKD
OPENMA
3, 8, 13
1
OPEN
OPENB
3, 8, 9, 13
OPENMB
8
OPENMY
1
OPEND
3, 6, 7, 9, 12, 13
OPENC
3, 9, 13
OPENMC
3, 13
OPENMD
6

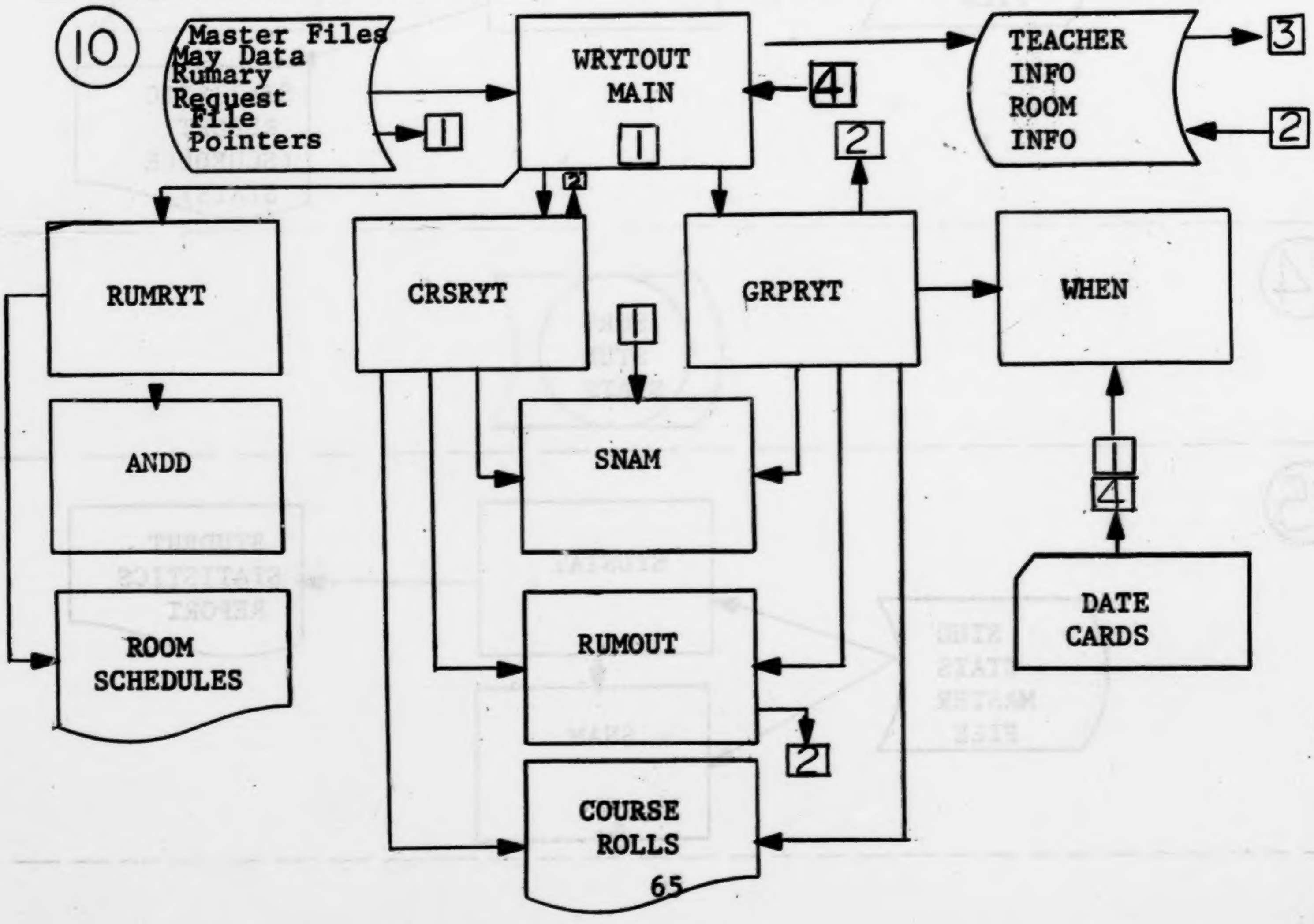
Over for Subroutines 1-13 and 17

DDCS SYSTEM FLOW AND LINK MAP

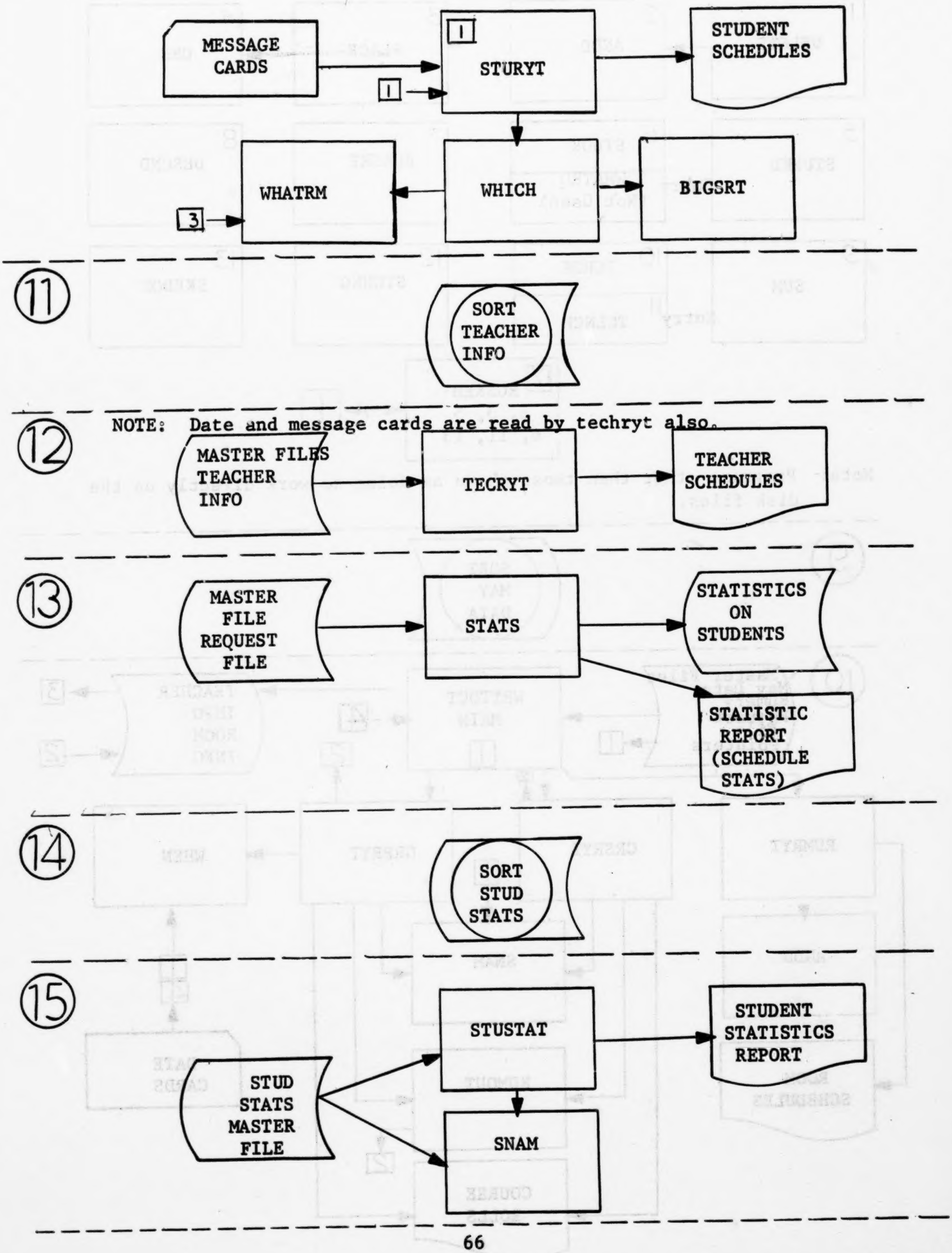

1
DELETE
2
ANDD
3
PLACE
4
ORR
5
STUSKD
6
STDOK
WHATRQ
(Not Used)
Entry
7
BLKSRT
8
DESCND
9
SUM
10
TCHOK
11
TCLNCH
Entry
12
STCHNG
13
SKEDOK
17
RGSKED
2, 3, 5
6, 11, 13
1

Note: Programs other than those shown as doing so work directly on the disk files.

9
SORT
MAY
DATA


10
Master Files
May Data
Rumary
Request
File
Pointers
1
WRYTOUT
MAIN
1
4
2
2
TEACHER
INFO
ROOM
INFO
3
2
RUMRYT
CRSRYT
GRPRYT
WHEN
1
ANDD
SNAM
1
4
ROOM
SCHEDULES
RUMOUT
DATE
CARDS
2
COURSE
ROLLS

DDCS SYSTEM FLOW AND LINK MAP


MESSAGE CARDS
1
STURYT
1
STUDENT SCHEDULES
WHATRM
3
WHICH
BIGSRT
11
SORT TEACHER INFO
12
MASTER FILES TEACHER INFO
TECRYT
TEACHER SCHEDULES
13
MASTER FILE REQUEST FILE
STATS
STATISTICS ON STUDENTS
STATISTIC REPORT (SCHEDULE STATS)
14
SORT STUD STATS
15
STUD STATS MASTER FILE
STUSTAT
SNAM
STUDENT STATISTICS REPORT

NOTE: Date and message cards are read by techryt also.

Appendix B

Daily Demand Computer Scheduling Program IBM 360/50
Print-Out Computer Program

Job Step 1

MAKE MASTER

```
C         THIS PROGRAM RESERVS THE FULL SPACE ON THE DISK FOR THE MASTER FILES
      IMPLICIT INTEGER*2 (A-Z)
      DIMENSION A(500),B(564),C(1025),D(70)
      DATA A,B,C,D/500*1,564*2,1025*3,70*4/
      DATA AV1,AV2,AV3,AV4,AV7,AV10/6*1/
      DEFINE FILE 1(8,250,U,AV1)
      DEFINE FILE 2(4,250,U,AV2)
      DEFINE FILE 3(54,282,U,AV3)
      DEFINE FILE 4(11,282,U,AV4)
      DEFINE FILE 7(1,2050,L,AV7)
      DEFINE FILE 10(1100,35,U,AV10)
      DO 10 I=1,8
   10 WRITE (1'AV1) A
      DO 20 I=1,4
   20 WRITE (2'AV2) A
      DO 30 I=1,54
   30 WRITE (3'AV3) B
      DO 40 I=1,11
   40 WRITE (4'AV4) B
   50 WRITE (7'AV7) C
      DO 60 I=1,1100
   60 WRITE (10'AV10) D
      CALL EXIT
      END
```

Job Step 2

BUILDER MAIN

```
C  THIS IS THE MAIN PROGRAM FOR BUILDING THE FILES
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON/BLOK03/$FILE,CHECK,DIGIT,NAMES,NLNGTH,SZFILE,NMCARD,ASVAR
      LOGICAL FLAG/T/, FLAGG/F/, FLAGR, FLAGS, FLAGT, STOP/F/, FLAGX/F/,
     1FLAGY/T/
      INTEGER*4 CARD(10), TITLE, GTITLE, COURSE, GRADE, ROOM, STUDNT,
     1TEACH,CHECK
      DIMENSION GRCARD(5), FILE 7(1025), GRFILE(24), WKAREA(2521),
     1SFILE(2500), DUMMY(2), WTEACH(10)
      EQUIVALENCE (NMCARD,NMRCRD)
      EQUIVALENCE (TITLE, CARD(1)), (CARD(10), DUMMY), (GTITLE, GRCARD),
     1(FILE 7(1002), GRFILE), (GRCARD(3), YEAR), (GRCARD(4), START),
     2(GRCARD(5),END), (WKAREA(22), SFILE), (WKAREA(21), COUNTS),
     3(DUMMY(2), CNUM), (WKAREA(11), WTEACH), (ASVAR, AV 1, AV 2, AV 3,
     4 AV4)
      DATA COURSE/'COUR'/, GRADE/'GRAD'/, ROOM/'ROOM'/, STUDNT/'STUD'/,
     1TEACH/'TEAC'/
C
C
 1000 FORMAT (2A4, A1, 6A4, 23X, I4, 20X)
 1001 FORMAT (2A2, 2X, I4, 2(6X, I4))
 1010 FORMAT (' FIRST CARD NOT COURSE CARD, ERROR CARDS IGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1012 FORMAT(' ONLY COURSE, TEACHER, ROOM, OR STUDENT CARDS ALLOWED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1014 FORMAT(' ONLY 2,500 STUDENT CARDS ALLOWED PER COURSE, ALL OVER 2,5
     A00 IGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1016 FORMAT (' STUDENT ALL CARDS NOT ALLOWED IN MASTER DECK, CARD IGNOR
     AED'/)
 1018 FORMAT (' ONLY 10 ROOM CARDS ALLOWED PER COURSE, ALL OVER 10 IGNOR
     AED'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1020 FORMAT (' ONLY 10 TEACHER CARDS ALLOWED PER COURSE, ALL OVER 10 IG
     ANORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1024 FORMAT (' GRADE CARDS MUST BE FIRST CARD IN DECK, ERROR CARDS IGNO
     ARED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
      DEFINE FILE  1(8,250,U,AV 1)
      DEFINE FILE  2(4,250,U,AV 2)
      DEFINE FILE  3(54,282,U,AV 3)
      DEFINE FILE  4(11,282,U,AV 4)
      DEFINE FILE  7(1,2050,L,AV 7)
      DEFINE FILE10(1100,35,U,AV10)
C
C
C
C     THIS PROGRAM BUILDS A DIRECT ACCESS FILE OF INFORMATION ABOUT THE COURSE
C     ( TEACHER, ROOM, AND STUDENT NUMBERS).  THE TEACHER AND STUDENT NUMBERS
C     ARE PUT IN ASCENDING ORDER, THE ROOM NUMBERS ARE IN ORDER OF PREFERENCE.
C
C     A FILE IS ALSO BUILT FOR TYPE OF NAME  STUDENT, TEACHER, COURSE, AND ROOM
C
C     A FILE IS BUILT THAT CONTAINS THE STARTING AND ENDING STUDENT NUMBERS FOR
C     GRADES 1-12.
C
C     FILE 15 IS USED TO STORE THE BEGINING LOCATION OF EACH COURSE, THE FIRST
C     BLANK RECORD, AND THE GRADE INFORMATION.
C
C
C                    ***GRADE  FILE***
C     THIS SECTION STORES THE GRADE INFORMATION IN BYTES 2003- 2050 OF FILE 15.
C
```

```
    5 DO 7 I=1,1025
    7 FILE 7(I)=0
   10 READ(8,1001,END=100) GRCARD
      IF(GTITLE.EQ.GRADE)GO TO 20
C     CHECK FOR A CARD OTHER THAN A GRADE CARD.  PRINT ONE ERROR IF NEEDED.
      IF(FLAGG) GO TO 10
      WRITE(6,1024)
      FLAGG= .TRUE.
      GO TO 10
   20 GRFILE(YEAR*2 -1)= START
      GRFILE(YEAR *2)= END
      GO TO 10
C
C
C                              ***COURSE  FILE***
C
C     THIS SECTION BUILDS FILE 10 WHICH CONTAINS INFORMATION ABOUT EACH COURSE
C
C
  100 AV10=1
  105 FLAGS=.FALSE.
      FLAGT= .FALSE.
      FLAGR= .FALSE.
      COUNTT= 0
      COUNTR= 0
      DO 107 I=1,21
  107 WKAREA(I)= 0
  110 READ(8,1000,END=261) CARD
C     CHECK TYPE OF CARD, PRINT ERROR IF NEEDED
      IF(FLAG) GO TO 120
      IF(TITLE.EQ.STUDNT) GO TO 200
      IF(TITLE.EQ.ROOM) GO TO 220
      IF(TITLE.EQ.TEACH) GO TO 240
  120 IF(TITLE.EQ.COURSE) GO TO 260
      IF(.NOT.FLAG) GO TO 140
      IF(FLAG.AND.FLAGY) GO TO 150
      GO TO 110
  140 IF(FLAGX) GO TO 110
      WRITE(6,1012) CARD
      FLAGX= .TRUE.
      GO TO 110
  150 WRITE(6,1010) CARD
      FLAGY=.FALSE.
      GO TO 110
C
C
C     STUDENT INFORMATION PROCESSED HERE
C
  200 COUNTS= COUNTS+1
      IF(CNUM.EQ.9999) GO TO 215
      IF(COUNTS.GE.2501) GO TO 210
      SFILE(COUNTS)= CNUM
C     CHECK FOR STUDENT NONE CARD, SET NUMBER OF STUDENTS WANTED TO ZERO
      IF(CNUM.EQ.0) COUNTS=0
      GO TO 110
C
C     IF TOO MANY STUDENT CARDS- ERROR PRINTED
  210 COUNTS= COUNTS-1
      IF(FLAGS) GO TO 110
      FLAGS= .TRUE.
      WRITE(6,1014)  CARD
      GO TO 110
C
C     IF STUDENT ALL CARD INSERTED= ERROR PRINTED
  215 COUNTS= COUNTS-1
      WRITE(6,1016) CARD
      GO TO 110
```

```
C
C
C     ROOM INFORMATION PROCESSED HERE
C
  220 COUNTR= COUNTR+1
      IF(COUNTR.GE.11) GO TO 230
      WKAREA(COUNTR)= CNUM
      GO TO 110
C
C     ERROR PRINTED IF TOO MANY COURSE CARDS
  230 COUNTR= COUNTR-1
      IF(FLAGR) GO TO 110
      FLAGR=.TRUE.
      WRITE(6,1018) CARD
      GO TO 110
C
C
C     TEACHER INFORMATION PROCESSED HERE
C
  240 COUNTT= COUNTT+1
      IF (CNUM.EQ.0) GO TO 245
      IF(COUNTT.GE.11) GO TO 250
      WTEACH(COUNTT)= CNUM
      GO TO 110
C
C     STAFF CARD IGNORED IF ONE FOUND
  245 COUNTT= COUNTT- 1
      GO TO 110
C
C     ERROR PRINTED IF TOO MANY TEACHER CARDS
  250 COUNTT= COUNTT- 1
      IF (FLAGT) GO TO 110
      FLAGT= .TRUE.
      WRITE (6,1020) CARD
      GO TO 110
C
C     WHEN  COURSE CARD READ TRANSFER HERE
  260 IF(.NOT.FLAG) GO TO 262
C     TRANSFER HERE IF IT IS THE FIRST COURSE CARD
      FLAG= .FALSE.
      FILE 7(CNUM)= AV10
      GO TO 110
C
C     TRANSFER HERE IF IT IS THE LAST CARD
  261 STOP= .TRUE.
C
C     ARRANGE TEACHERS IN ORDER AND CHECK FOR DUPLICATIONS
  262 TEMP= COUNTT
      CALL SORT (WTEACH, COUNTT)
      CALL DUPCHK(WTEACH, COUNTT)
      IF(TEMP.EQ.COUNTT) GO TO 270
      A= COUNTT+1
      DO 265 I= A, TEMP
  265 WTEACH(I)= 0
C
C     ARRANGE STUDENTS IN ORDER AND CHECK FOR DUPLICATIONS
  270 CALL SORT (SFILE, COUNTS)
      CALL DUPCHK (SFILE, COUNTS)
C
C     WRITE THE INFORMATION ABOUT THE COURSE ON DISK
      A=COUNTS +21
      WRITE (10'AV10) (WKAREA(I), I=1,A)
      IF (STOP) GO TO 280
C
C     IF NOT LAST COURSE TRANSFER HERE
      FILE 7(CNUM)= AV10
      GO TO 105
C
C     WRITE FILE 15 AFTER THE LAST COURSE                  ***R*********
  280 FILE 7(1001)= AV10
      AV 7= 1
      WRITE( 7'AV 7) FILE 7
```

```
C
C
C
C
C                             ***NAME FILES***
C
C     PREPARE VARIABLES FOR WORK ON VARIOUS NAME FILES
C
C
C                             COURSE NAME FILE
C
  400 $FILE=  1
      CHECK= COURSE
      DIGIT= 1
      NAMES= 125
      NLNGTH= 2
      SZFILE=250
      NMRCRD= 8
      CALL NMFILE
C
C
C                             ROOM NAME FILE
C
  500 $FILE=  2
      CHECK= ROOM
      DIGIT= 2
      NAMES= 125
      NLNGTH= 2
      SZFILE=250
      NMRCRD= 4
      CALL NMFILE
C
C
C                             STUDENT NAME FILE
C
  600 $FILE=  3
      CHECK= STUDNT
      DIGIT= 3
      NAMES= 47
      NLNGTH= 6
      SZFILE= 282
      NMRCRD= 54
      CALL NMFILE
C
C
C                             TEACHER NAME FILE
C
  700 $FILE=  4
      CHECK= TEACH
      DIGIT= 4
      NAMES= 47
      NLNGTH= 6
      SZFILE= 282
      NMRCRD= 11
      CALL NMFILE
      CALL EXIT
      END
```

NMFILE

```
      SUBROUTINE NMFILE
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON/BLOK03/$FILE,CHECK,DIGIT,NAMES,NLNGTH,SZFILE,NMCARD,ASVAR
      LOGICAL FLAG, FLAGR, FLAGS, FLAGT
      INTEGER*4 CARD(10), SNAME(6),WKAREA(282),NONAME(6), TITLE, CHECK
      DIMENSION DUMMY(2)
      EQUIVALENCE (NMCARD,NMRCRD)
      EQUIVALENCE (TITLE, CARD(1)),(CARD(4), SNAME), (CARD(10),DUMMY),
     1(DUMMY(2), CNUM)
      DATA NONAME/'NO NAME                   '/
C
C
 1000 FORMAT (2A4, A1, 6A4, 23X, I4, 20X)
 1010 FORMAT (' ONLY COURSE CARDS ALLOWED IN COURSE NAME DECK, ERROR CAR
     ADS IGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1012  FORMAT (' ONLY ROOM CARDS ALLOWED IN ROOM NAME DECK, ERROR CARDS
     AIGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1014 FORMAT (' ONLY STUDENT CARDS ALLOWED IN STUDENT NAME DECK, ERROR C
     AARDS IGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1016 FORMAT (' ONLY TEACHER CARDS ALLOWED IN TEACHER NAME DECK, ERROR C
     AARDS IGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1020 FORMAT (' NAME CARDS MUST BE IN ASCENDING ORDER, ALL ERROR CARDS I
     AGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
 1022 FORMAT (' NAME CARDS WITH DUPLICATE NUMBERS NOT ALLOWED, ERROR CAR
     ADS IGNORED,'/
     12H '2A4, A1, 6A4, 23X, I4, 20X '''WAS FIRST CARD IGNORED,')
C
C
  100 ASVAR= 1
      LCNUM= 0
      FLAG = .FALSE.
      FLAGR= .FALSE.
      FLAGS= .FALSE.
      FLAGT= .FALSE.
      A= 0
  200 READ(8,1000,END= 250) CARD
      IF(TITLE.NE.CHECK .OR.CNUM.EQ.0 .OR.CNUM.EQ.9999) GO TO 210
  205 IF(CNUM.GT.A* NAMES) GO TO 220
      IF(CNUM.LT.LCNUM) GO TO 230
      IF(CNUM.EQ.LCNUM) GO TO 240
      B= (CNUM-(C* NAMES)) *NLNGTH -NLNGTH
C
C     PLACE NAME IN THE FILE BY ITS NUMBER
      DO 207 I= 1, NLNGTH
  207 WKAREA(B+I)= SNAME(I)
      LCNUM= CNUM
      FLAGT= .FALSE.
      FLAGS= .FALSE.
      GO TO 200
C
C     ERROR PRINTED IF WRONG TYPE OF NAME CARD
  210 IF(FLAG) GO TO 200
      FLAG= .TRUE.
      IF (DIGIT.EQ.1) WRITE(6,1010) CARD
      IF (DIGIT.EQ.2) WRITE(6,1012) CARD
      IF (DIGIT.EQ.3) WRITE(6,1014) CARD
      IF (DIGIT.EQ.4) WRITE(6,1016) CARD
      GO TO 200
C
C     WRITE OUT RECORD IF NEEDED, PREPARE WORKING AREA FOR NEW NAMES
  220 ASSIGN 205 TO $WITCH
  221 IF (FLAGR) WRITE($FILE'ASVAR)(WKAREA(I),I= 1,SZFILE)
      DO 225 I= 1, SZFILE, NLNGTH
      J= I-1
      DO 225 K= 1, NLNGTH
  225 WKAREA(J+K)= NONAME(K)
```

```
      C=A
      A= A+1
      FLAGR= .TRUE.
        GO TO $WITCH,(205,260)
C
C     ERROR PRINTED IF CARDS ARE IN WRONG ORDER
  230 IF(FLAGT) GO TO 200
      FLAGT= .TRUE.
      WRITE(6,1020) CARD
      GO TO 200
C
C     ERROR PRINTED IF NAME NUMBERS ARE EQUAL
  240 IF(FLAGS) GO TO 200
      FLAGS= .TRUE.
      WRITE(6,1022) CARD
      GO TO 200
C
C     FILL REMAINING FILE SPACE
  250 IF(A.GT.0) GO TO 255
      A= 1
      NMRCRD= NMRCRD+ 1
  255 F=A
      DO 260 M= F, NMRCRD
      ASSIGN 260 TO $WITCH
      GO TO 221
  260 CONTINUE
      RETURN
      END
```

SORT

```
      SUBROUTINE SORT(A,B)
C         THIS SUBROUTINE ORDERS THE ITEMS IN 'A' IN ASCENDING ORDER
C             'A' IS A SINGLE DIMENSIONED ARRAY. 'B' IS THE NUMBER OF ITEMS
C         THE ARRAY CONTAINS
C
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      DIMENSION A(1)
C
C
      IF(B .LT.2) RETURN
      N=B-1
      DO 10 J=1,N
      I=J
    9 IF(A(I).LE.A(I+1)) GO TO 10
      TEMP=A(I)
      A(I)=A(I+1)
      A(I+1)=TEMP
      I=I-1
      IF(I)10,10,9
   10 CONTINUE
      RETURN
      END
```

DUPCHK

```
      SUBROUTINE DUPCHK(A,NUM)
C
C         THIS SUBROUTINE CHECKS FOR AND ELIMINATES ANY DUPLICATION
C         IN ITEMS IN THE ARRAY SENT TO IT. IT ALSO SETS NUM TO THE NEW
C         NUMBER OF ITEMS IN THE ARRAY AFTER THE ARRAY HAS BEEN COMPACTED
C
C         'A' IS A SINGLE DIMENSIONED ARRAY. 'NUM' IS THE NUMBER OF ITEMS
C         IN 'A' AS IT ENTERS AND LEAVES THIS SUBROUTINE. 'N' IS NUM
C         (AS IT ENTERS), MINUS THE NUMBER OF DUPLICATIONS FOUND.
C         NOTE   'A' MUST CONTAIN AN ORDERED TABLE IF THIS  SUBROUTINE
C         IS TO FUNCTION PROPERLY.
C
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      DIMENSION A(1)
C
C
      IF(NUM .LT.2) RETURN
      DO 10 K=2,NUM
      IF(A(K).EQ.A(K-1)) GO TO 20
   10 CONTINUE
      RETURN
   20 N=K-1
      DO 30 J=K,NUM
      IF(A(N).NE.A(J)) N=N+1
   30 A(N)=A(J)
      NUM=N
      RETURN
      END
```

```
C THIS PROGRAM PRINTS OUT A COPY OF THE MASTER FILES
      IMPLICIT INTEGER *2 (A-Z)                                         TSTMSTE
      DIMENSION A(1025),B(70),C(500),D(564)                             TSTMSTE
      DEFINE FILE 1(8,250,U,AV1)                                        TSTMSTE
      DEFINE FILE 2(4,250,U,AV2)                                        TSTMSTE
      DEFINE FILE 3(54,282,U,AV3)                                       TSTMSTE
      DEFINE FILE 4(11,282,U,AV4)                                       TSTMSTE
      DEFINE FILE 7(1,2050,L,AV7)                                       TSTMSTE
      DEFINE FILE 10(1100,35,U,AV10)                                    TSTMSTE
  100 FORMAT (' ', 25I5)                                                TSTMSTE
  200 FORMAT (' ',60A2)                                                 TSTMSTE
  300 FORMAT ('1')                                                      TSTMSTE
      READ(7'1) A                                                       TSTMSTE
      WRITE(6,100) A                                                    TSTMSTE
      WRITE(6,300)
      DO 20 I=1,200                                                     TSTMSTE
      IF(A(I).EQ.0) GO TO 20                                            TSTMSTE
      READ(10'A(I)) B                                                   TSTMSTE
      WRITE (6,100) B                                                   TSTMSTE
    5 IF(B(21).LT.50) GO TO 20                                          TSTMSTE
      SAVE = B(21) - 49                                                 TSTMSTE
    8 READ(10'AV10) B                                                   TSTMSTE
      WRITE (6,100) B                                                   TSTMSTE
      SAVE = SAVE - 70                                                  TSTMSTE
      IF (SAVE.GT.0) GO TO 8                                            TSTMSTE
   20 CONTINUE                                                          TSTMSTE
      WRITE (6,300)                                                     TSTMSTE
      READ(1'1) C                                                       TSTMSTE
      WRITE(6,200) C                                                    TSTMSTE
      READ(1'2) C                                                       TSTMSTE
      WRITE(6,200) C                                                    TSTMSTE
      WRITE(6,300)                                                      TSTMSTE
      READ(2'1) C                                                       TSTMSTE
      WRITE(6,200) C                                                    TSTMSTE
      WRITE(6,300)                                                      TSTMSTER
      DO 40 I=1,13                                                      TSTMSTER
      READ(3'I) D                                                       TSTMSTER
   40 WRITE (6,200) D                                                   TSTMSTER
      WRITE(6,300)                                                      TSTMSTER
      DO 50 I=1,3                                                       TSTMSTER
      READ(4'I) D                                                       TSTMSTER
   50 WRITE(6,200) D                                                    TSTMSTER
      CALL EXIT                                                         TSTMSTER
      END
```

Job Step 3

COPYOUT

```
C THIS PROGRAM MAKES A BACK UP COPY OF THE MASTER FILES ON TAPE  COPYOUT
      IMPLICIT INTEGER*2(A-Z)
      DIMENSION A(500),B(564),C(1025),D(70)
      DATA AV1,AV2,AV3,AV4,AV7,AV10/6*1/
      DEFINE FILE 1(8,250,U,AV1)
      DEFINE FILE 2(4,250,U,AV2)
      DEFINE FILE 3(54,282,U,AV3)
      DEFINE FILE 4(11,282,U,AV4)
      DEFINE FILE 7(1,2050,L,AV7)
      DEFINE FILE 10(1100,35,U,AV10)
      DO 10 I=1,8
      READ(1'AV1) A
   10 WRITE(9) A
      DO 20 I=1,4
      READ(2'AV2) A
   20 WRITE (9) A
      DO 30 I=1,54
      READ(3'AV3) B
   30 WRITE (9) B
      DO 40 I=1,11
      READ(4'AV4) B
   40 WRITE(9) B
      READ(7'AV7) C
      WRITE(9) C
      K= C(1001)-1
      DO 60 I=1,K
      READ(10'AV10) D
   60 WRITE(9) D
      ENDFILE 9
      CALL EXIT
      END
```

Job Step 4

COPYIN

```
 C THIS PROGRAM COPIES THE MASTER FILES FROM TAPE TO DISK
       IMPLICIT INTEGER*2(A-Z)
       DIMENSION A(500),B(564),C(1025),D(70)
       DATA AV1,AV2,AV3,AV4,AV7,AV10/6*1/
       DEFINE FILE 1(8,250,U,AV1)
       DEFINE FILE 2(4,250,U,AV2)
       DEFINE FILE 3(54,282,U,AV3)
       DEFINE FILE 4(11,282,U,AV4)
       DEFINE FILE 7(1,2050,L,AV7)
       DEFINE FILE 10(1100,35,U,AV10)
       DO 10 I=1,8
       READ(9) A
  10   WRITE(1'AV1) A
       DO 20 I=1,4
       READ(9) A
  20   WRITE(2'AV2) A
       DO 30 I=1,54
       READ(9) B
  30   WRITE(3'AV3) B
       DO 40 I=1,11
       READ(9) B
  40   WRITE(4'AV4) B
       READ(9) C
       WRITE(7'AV7) C
       K= C(1025)-1
       DO 60 I= 1,K
       READ(9) D
  60   WRITE(10'AV10) D
       [illegible]
       CALL EXIT
```

Job Step 5

UPDATE

```
C      MASTER UPDATE 'MAIN'
C
C                                               ************************
C                                               * PROGRAM DESCRIPTION *
C                                               ************************
C
C         THE MASTER UPDATE SYSTEM CONSISTS OF SEVEN SUBROUTINES AND
C         A MAIN PROGRAM.  SUBROUTINES ADDMOD,DELMOD AND CHAMOD
C         MAKE ADDITIONS,DELETIONS AND CHANGES TO THE COURSE FILE.
C         SUBROUTINE NAMCON MAKES ALL ALTERATIONS TO THE NAME FILES.
C         SUBROUTINE SORT SORTS ARRAYS IN ASCENDING ORDER.
C         SUBROUTINE TWOCHK ELIMINATES DUPLICATE NUMBERS FROM AN
C         UNSORTED ARRAY.  SUBROUTINE DUPCHK DOES THE SAME AS TWOCHK
C         WITH SORTED ARRAYS
C
C                                               **************
C                                               * DICTIONARY *
C                                               **************
C
C         MASLOK IS USED TO HOLD THE POINTERS FOR THE COURSE FILE
C         CARD IS THE ARRAY ALL INPUT CARDS ARE READ INTO
C         AV10 IS THE ASSOCIATED VARIABLE FOR THE COURSE FILE
C         CONTAG IS USED TO CAUSE ONLY THE FIRST CARD IN ERROR TO BE
C         PRINTED.  ADD,DEL,CHA ARE USED AS CHECKS FOR CONTROL CARDS.
C         FN IS THE FILE NUMBER WHEN NAMCON IS CALLED
C         F1 SHOWS WHICH NAME FILE IS TO BE WORKED ON
C         F2,F3,F4,F5 ARE DO LOOP PARAMETERS  USED BY NAMCON
C         SEE DOCUMENTATION IN NAMCON FOR THE DETAILS
C         XY IS ONE MORE THAN THE NUMBER OF NAMES PER RECORD
C         F7 IS THE NUMBER OF RECORDS IN THE FILE
C         OF THE FILE TO BE WORKED ON BY NAMCON
C
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      INTEGER*4 FN
C
      COMMON/BLOK04/MASLOK(1025),CARD(17),AV10
C
      LOGICAL*1 CONTAG/.FALSE./
C
      DATA ADD,DEL,CHA/'AD','DE','CH'/
C
C
 1000 FORMAT(4A2,1X,12A2,T57,I4)
 1100 FORMAT('0',T10,'COURSE FILE CHANGES ERROR '/,' ','ILLEGAL CARD',
     1' ENCOUNTERED. FIRST CARD MUST BE AN ADD,DELETE OR CHANGE CARD.',
     2' CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
C
C         FILE 7 IS THE POINTERS FILE.  FILE 10 IS THE COURSE FILE
C
      DEFINE FILE 10(1100,35,U,AV10)
      DEFINE FILE 7(1,2050,L,AV7)
C
C
C                                               ********************
C                                               * READ FIRST CARD *
C                                               ********************
C
      READ(8,1000,END=100) CARD
C
C                                               ************************
C                                               * READ POINTERS FILE *
C                                               ************************
C
```

```
C
      READ(7'1) MASLOK
C
C
C                                        *************
C                                        * BRANCHING *
C                                        *************
C
C        * CHECK THE CARD AND BRANCH TO APPROPRIATE SUBROUTINE *
C
 10   IF(CARD(1).EQ.ADD) CALL ADDMOD(+11,+100)
  11  IF(CARD(1).EQ.DEL) CALL DELMOD(+10,+100)
      IF(CARD(1).EQ.CHA) CALL CHAMOD(+10,+100)
C
C                                        *****************
C                                        * ERROR ROUTINE *
C                                        *****************
C
C        CHECK FOR A PREVIOUS ERROR BRANCH TO READ IF THERE HAS BEEN
      IF(CONTAG) GO TO 12
C
C        PRINT OUT MESSAGE AND CARD IN ERROR
      WRITE(6,1100) CARD
C
C        SET THE CONTROL TO AVOID FUTURE PRINT OUT
      CONTAG=.TRUE.
C
C        READ A CARD AND BRANCH BACK TO CHECK IT OUT
 12   READ(8,1000,END=100) CARD
      GO TO 10
C
C
C                                        ***************
C                                        * COURSE FILE *
C                                        ***************
C
C        START WORKING ON THE COURSENAME FILE.
C
C        READ THE FIRST CARD TO CHECK FOR WORK ON THIS FILE
 100  READ(8,1000,END=200) CARD
C
C
C                                        **************
C                                        * INITIALIZE *
C                                        **************
C
C        INITIALIZE THE ARGUMENTS
      F1=1
      F2=5
      F3=9
      F4=8
      XY=126
      F5=4
      FN=1
      F7=4
      CALL NAMCON (F1,F2,F3,F4,XY,F5,FN,F7)
C
C
C                                        ******************
C                                        * ROOM NAME FILE *
C                                        ******************
C
C        START WORKING ON THE ROOMNAME FILE.
C        READ THE FIRST CARD TO CHECK FOR WORK ON THIS FILE
C
 200  READ(8,1000,END=300) CARD
C
C
C                                        **************
C                                        * INITIALIZE *
C                                        **************
C
```

```
C         INITIALIZE THE ARGUMENTS
      F1=2
      F2=5
      F3=9
      F4=8
      XY=126
      F5=4
      FN=2
      F7=2
      CALL NAMCON (F1,F2,F3,F4,XY,F5,FN,F7)
C
C                                              ****************
C                                              * STUDENT NAME *
C                                              *    FILE      *
C                                              ****************
C
C
C         START WORKING ON THE STUDENTNAME FILE.
C         READ THE FIRST CARD TO CHECK FOR WORK ON THIS FILE
C
  300 READ(8,1000,END=40C) CARD
C
C                                              **************
C                                              * INITIALIZE *
C                                              **************
C
C
C         INITIALIZE THE ARGUMENTS
      F1=3
      F2=13
      F3=17
      F4=16
      XY=48
      F5=12
      FN=3
      F7=54
      CALL NAMCON (F1,F2,F3,F4,XY,F5,FN,F7)
C
C                                              ****************
C                                              * TEACHER NAME *
C                                              *    FILE      *
C                                              ****************
C
C
C         START WORKING ON THE TEACHERNAME FILE.
C         READ THE FIRST CARD TO CHECK FOR WORK ON THIS FILE
C
  400 READ(8,1000,END=500) CARD
C
C                                              **************
C                                              * INITIALIZE *
C                                              **************
C
C
C         INITIALIZE THE ARGUMENTS
      F1=4
      F2=13
      F3=17
      F4=16
      XY=48
      F5=12
      FN=4
      F7=11
      CALL NAMCON (F1,F2,F3,F4,XY,F5,FN,F7)
C
C                                              ************
C                                              * SIGN OFF *
C                                              ************
C
  500 CALLEXIT
      END
```

ADDMOD

```
      SUBROUTINE ADDMOD(*,*)
C
C        THE PURPOSE OF THIS SUBROUTINE IS TO MAKE ADDITIONS TO THE
C        COURSE FILE.
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON/BLOK04/MASLOK(1025),CARD(17),AV10
      COMMON /BLOK06/ SNAME(1200),TNAME(120),RNAME(40),TEMFIL(121),
     1CFILE(2521)
      COMMON /BLOK14/ OVER
      LOGICAL *1 FINDER,CHECK,STAG,RTAG,TTAG,RERR,TERR,SERR,CONTAG,
     1ADDTAG,DELTAG,CHATAG
      DIMENSION TEMFI1(10), TEMFI2(10), TEMFI3(100)
      DIMENSION  CFILE1(10),CFILE2(10),CFILE3(2500)
      DIMENSION CNAME(5)
      EQUIVALENCE (TEMFIL(1),TEMFI1(1)),(TEMFIL(11),TEMFI2(1))
      EQUIVALENCE( TEMFIL(22),TEMFI3(1))
      EQUIVALENCE (CFILE(1),CFILE1(1)),(CFILE(11),CFILE2(1))
      EQUIVALENCE (CFILE(22),CFILE3(1))
      DATA CORS,STUD,TECH,ROOM/'CO','ST','TE','RO'/
      DATA ADD,DEL,CHA/'AD','DE','CH'/
C
C
 1000 FORMAT(4A2,1X,12A2,T57,I4)
 1100 FORMAT('0',T10,'COURSE FILE ADD ERROR'/' ',T10,'THE',
     1' CARD FOLLOWING AN ADD, DELETE,OR CHANGE CARD MUST BE A COURSE
     3CARD. CARD IN ERROR READS AS FOLLOWS'/,' ',T10,4A2,1X,12A2,22X,I4)
 1102 FORMAT('0',T10,'COURSE FILE ADD ERROR'/,' ',T10,
     1'ILLEGAL CARD ENCOUNTERED.',2X,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1103 FORMAT('0',T10,'COURSE FILE ADD ERROR'/,' ',T10,
     1'TOO MANY STUDENTS WERE TO BE ADDED ALL OTHERS TO BE ADDED INCLUDI
     2NG THE FOLLOWING CARD WILL BE IGNORED.'/,' ',T10,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1104 FORMAT('0',T10,'COURSE FILE ADD ERROR'/,' ',T10,
     1'TOO MANY TEACHERS WERE TO BE ADDED. ALL OTHERS TO BE ADDED INCLUD
     2ING THE FOLLOWING CARD WILL BE IGNORED.'/,' ',T10,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1105 FORMAT('0',T10,'COURSE FILE ADD ERROR'/,' ',T10,
     1'TOO MANY ROOMS WERE TO BE ADDED ALL OTHERS TO BE ADDED INCLUDING
     2THE FOLLOWING CARD WILL BE IGNORED.'/,' ',T10,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1106 FORMAT('0',T10,'COURSE FILE ADD WARNING'/,' ',T10,
     1'NO ROOMS MAY BE ADDED TO COURSE',2X,4A2,2X,I4)
 1107 FORMAT('0',T10,'COURSE FILE ADD WARNING'/,' ',T10,
     1'NO MORE ROOMS MAY BE ADDED TO COURSE  ',4A2,2X,I4/,' ',T10,
     2 I4,'ROOMS WERE ADDED THIS UPDATE ')
 1108 FORMAT('0'T20,'THE FOLLOWING ROOMS WERE ADDED TO COURSE  ',4A2,I4)
 1109 FORMAT('0',T2,10(4A2,I4,1X))
```

```
1110 FORMAT('0',T10,'COURSE FILE ADD WARNING'/,' ',T10,
    1'NO TEACHERS MAY BE ADDED TO COURSE',2X,4A2,2X,I4)
1111 FORMAT('0',T10,'COURSE FILE ADD WARNING'/,' ',T10,
    1'NO MORE TEACHERS MAY BE ADDED TO COURSE  ',4A2,2X,I4,/,' ',T10,
    2 I4, 'TEACHERS WERE ADDED THIS UPDATE.')
1112 FORMAT('0',/,4(2X,12A2,2X,I4))
1114 FORMAT('0',T20,'THE FOLLOWING TEACHERS HAVE BEEN ADDED TO COURSE',
    12X,4A2,I4)
1115 FORMAT('0',T20,'THE FOLLOWING STUDENTS HAVE BEEN ADDED TO COURSE',
    12X,4A2,I4)
1116 FORMAT('0',T30,'COURSE FILE  UPDATE HAS BEEN ABORTED BECAUSE  THE
    1NUMBER OF RECORDS USED NOW EXCEEDS', I5)
1117 FORMAT('0',T10,'COURSE FILE ADD ERROR'/,' ',T10,
    1'TWO COURSE CARDS TOGETHER. THE SECOND CARD WILL BE USED.'/,' ',
    2T10,'THIS CARD READS AS FOLLOWS'/,' ',T10,4A2,1X,12A2,22X,I4)
C
C
C         STATEMENTS 0 THRU 10-1 INITIALIZATION OF VARIABLES.
C
    4 ASSIGN 10 TO $IN
    1 DO 2 I=1,121
    2 TEMFIL(I)=0
      CONTAG=.FALSE.
      CHECK=.FALSE.
      STAG=.FALSE.
      RTAG=.FALSE.
      TTAG=.FALSE.
      SERR=.FALSE.
      TERR=.FALSE.
      RERR=.FALSE.
      SC=0
      TC=0
      RC=0
      ADDTAG=.FALSE.
      CHATAG=.FALSE.
      DELTAG=.FALSE.
      GO TO $IN,(10,20)
C
C         STATEMENT 10 READS IN FIRST CARD.
 10   READ(8,1000,END=901) CARD
C
C         CHECK FOR COURSE CARD.
      IF(CARD(1).EQ.CORS) GO TO 20
C
C         CHECK FOR ADDITION, DELETION OR CHANGE CARD.
      IF(CARD(1).EQ.ADD) GO TO 10
      IF(CARD(1).EQ.DEL.OR.CARD(1).EQ.CHA) RETURN 1
C
C         ERROR--FIRST CARD WAS NOT A COURSE CARD.
      IF(CONTAG) GO TO 10
      WRITE(6,1100) CARD
      CONTAG=.TRUE.
      GO TO 10
C
```

```
C         LOCATE THE COURSE RECORD.
C         IF NO COURSE RECORD EXISTS, LOCATE THE FIRST OPEN RECORD.
   20 AV10=MASLOK(CARD(17))
      FINDER=AV10.EQ.0
      IF(FINDER) AV10= MASLOK(1001)
      FIND(10'AV10)
C
C         STORE COURSE NAME AND NUMBER.
      DO 25  I =1,4
   25 CNAME(I)= CARD(4+I)
      CNAME(5)= CARD(17)
C
C         STATEMENT 12 IS USED TO READ IN ALL ADDITIONAL CARDS.
   12 READ(8,1000,END=505) CARD
C
C         CHECK FOR STUDENT, TEACHER, ROOM, OR COURSE CARD.
      IF(CARD(1).EQ.STUD) GO TO 200
      IF(CARD(1).EQ.TECH) GOTO 300
      IF(CARD(1).EQ.ROOM) GO TO 400
      IF(CARD(1).EQ.CORS) GO TO 500
C
C         CHECK FOR ADDITION, DELETION OR CHANGE CARD.
      ADDTAG= CARD(1).EQ.ADD
      DELTAG= CARD(1).EQ.DEL
      CHATAG= CARD(1).EQ.CHA
      IF(ADDTAG.OR.CHATAG.OR.DELTAG) GO TO  600
C
C         ERROR--ILLEGAL CARD ENCOUNTERED.
      WRITE(6,1102) CARD
      GO TO 12
C
C         STATEMENTS 200 THRU 300-1 ARE USED TO PROCESS STUDENT CARDS.
C         CHECK FOR STUDENT OVERFLOW. (LIMIT IS 100)
  200 IF(SERR) GO TO 12
C
C         CHECK FOR STUDENTS NONE OR STUDENTS ALL CARD.  THESE CARDS ARE
C         IGNORED.
      IF(CARD(17).EQ.0.OR.CARD(17).EQ.9999) GO TO 12
C
C         INDICATOR SHOWING STUDENT (S) ADD IS SET .TRUE.
  201 STAG=.TRUE.
C
C         SC CONTAINS THE NUMBER OF STUDENTS ADDED.
      SC=SC+1
C
C         STUDENT OVERFLOW CHECK.
      IF(SC.EQ.101) GO TO 220
C
C         STORE TOTAL OF STUDENTS ADDED.
      TEMFIL(21)= SC
C
```

```
C         STORE STUDENT NAME AND NUMBER.
      TEMFI3(SC)= CARD(17)
      K=SC*12-16
      DO 202 I=5,16
  202 SNAME(K+I)= CARD(I)
      GO TO 12
C
C         STUDENT OVERFLOW INDICATOR SET .TRUE.
C         ERROR--STUDENT OVERFLOW MESSAGE.
  220  SERR=.TRUE.
      WRITE(6,1103) CARD
      SC=100
      GO TO 12
C
C         STATEMENTS 300 THRU 400-1 PROCESS TEACHER CARDS.
C         PROCESSING IS THE SAME FOR TEACHERS AS FOR STUDENTS.
  300 IF(TERR) GO TO 12
      TTAG=.TRUE.
      TC=TC+1
      IF(TC.EQ.11) GO TO 320
      TEMFI2(TC)=CARD(17)
      K=TC*12-16
      DO 302 I=5,16
  302 TNAME(K+I)= CARD(I)
      GO TO 12
  320 TERR=.TRUE.
      WRITE(6,1104) CARD
      TC=10
      GO TO 12
C
C         STATEMENTS 400 THRU 500-1 PROCESS ROOM CARDS.
C         PROCESS IS THE SAME FOR ROOMS AS FOR STUDENTS.
  400 IF(RERR) GO TO 12
      RC=RC+1
      IF(RC.GE.11) GO TO 420
      RTAG=.TRUE.
      TEMFI1(RC)=CARD(17)
      K=RC*4-8
      DO 402 I=5,8
  402 RNAME(K+I)= CARD(I)
      GO TO 12
  420 RERR=.TRUE.
      WRITE(6,1105) CARD
      RC=10
      GO TO 12
C
C         ERROR CHECK--TWO COURSE CARDS TOGETHER.
  500 IF(STAG.OR.RTAG.OR.TTAG) GO TO 600
      WRITE(6,1117)CARD
      GO TO 20
C
C         CHECK FOR END OF COURSE FILE ALTERATIONS.
  505 IF(.NOT.(STAG.OR.RTAG.OR.TTAG)) GO TO 901
      CHECK=.TRUE.
```

```
C
C         CHECK FOR NEW COURSE.
  600 IF(FINDER) GO TO 800
C
C         READ IN FIRST OLD RECORD.
      READ(10'AV10) (CFILE(I), I=1,70)
C
C         STATEMENTS 604 THRU 650-1 MAKE ROOM ADDITIONS AND PRINT OUT THE
C         ROOMS ACTUALLY ADDED.
C         CHECK TO SEE IF ROOMS WILL BE ADDED.
  604 IF(.NOT.RTAG) GO TO 650
C
C         LOCATE FIRST OPENING IN ROOMS LIST.
      DO 601 IN=1,10
      IF(CFILE1(IN).EQ.0) GO TO 602
  601 CONTINUE
C
C         MESSAGE--NO ROOM OPENINGS.
      WRITE(6,1106) CNAME
      GO TO 650
  602 R2=RC
C
C         LOAD IN NEW ROOMS.
      DO 603 OUT= IN,10
  603 CFILE(OUT)= TEMFI1(OUT-IN+1)
C
C         CHECK FOR DUPLICATIONS.
      CALL TWOCHK (CFILE1)
C
C         CHECK FOR OVERFLOW. (LIMIT IS 10)
      IF(RC.GT.11-IN) RC=11-IN
C
C         PRINT OUT OVERFLOW MESSAGE.
      IF(RC.NE.R2) WRITE(6,1107) CNAME,RC
C
C         PRINT OUT OF ROOMS ADDED.
      WRITE(6,1108) CNAME
      WRITE(6,1109)(((RNAME(X*4-4+I),I=1,4),TEMFI1(X)),X=1,RC)
C
C         STATEMENTS 650 THRU 659 MAKE TEACHER ADDITIONS AND WRITE OUT
C         TEACHERS ACTUALLY ADDED.
C         PROCESS IS THE SAME AS FOR ROOMS.
  650 IF(.NOT.TTAG) GO TO 670
      DO 651 IN=1,10
      IF(CFILE2(IN).EQ.0) GO TO 652
  651 CONTINUE
      WRITE (6,1110) CNAME
      GO TO 670
  652 R2=TC
      DO653 OUT=IN,10
  653 CFILE2(OUT)= TEMFI2(OUT-IN+1)
      IF(TC.GT.11-IN) TC=11-IN
      R2=IN+TC
      CALL SORT(CFILE2,R2)
      CALL DUPCHK(CFILE2,R2)
      IF(R2.NE.IN+TC) WRITE(6,1111) CNAME,TC
      WRITE(6,1114) CNAME
  659 WRITE(6,1112)(((TNAME(X*12-12+I),I=1,12),TEMFI2(X)),X=1,TC)
C
C          CALCULATE THE NUMBER OF RECORDS USED BY THE COURSE BEING
C          PROCESSED BEFORE ADDITIONS WERE MADE.
  670 M= ((CFILE(21)+20)/70)+1
C
C          STATEMENTS 670+1 THRU 700-1 MAKE STUDENT ADDITIONS AND PRINT
C          OUT THE STUDENTS ADDED.
C
C          CHECK TO SEE IF STUDENTS WILL BE ADDED.
      IF(.NOT.STAG) GO TO 700
C
C          CHECK TO SEE IF MORE OLD RECORDS NEED TO COME IN.
      IF(CFILE(21).LE.49) GO TO 671
      IN= CFILE(21)
C
C          READ IN THE REMAINDER OF THE RECORDS OF THE COURSE BEING
C          PROCESSED.
      READ(10'AV10)(CFILE3(I),I=50,IN)
  671 IN= CFILE(21)
C
C          ADD IN THE NEW STUDENTS.
      DO 702 I=1,SC
  702 CFILE3(IN+I)=TEMFI3(I)
      R2=IN+SC
C
C          SORT THE OLD STUDENR NUMBERS COMBINED WITH THE NEW
      CALLSORT(CFILE3,R2)
C
C          CHECK FOR NUMBER DUPLICATIONS.
      CALLDUPCHK(CFILE3,R2)
C
C          STORE NUMBER OF STUDENTS NOW IN THE COURSE.
      CFILE(21)=R2
C
C          PRINT OUT OF STUDENTS ADDED.
      WRITE(6,1115) CNAME
      WRITE(6,1112)(((SNAME(X*12-12+I),I=1,12),TEMFI3(X)),X=1,SC)
C
C          SET UP ASSOCIATEDVARIABLE FOR WRITE OUT OF COURSE RECORDS.
  700 AV10= MASLOK(CNAME(5))
      IF(STAG.AND.M.NE.((CFILE(21)+20)/70)+1) AV10= MASLOK(1001)
      M=CFILE(21)+21
      IF(AV10.EQ.MASLOK(1001))MASLOK(CNAME(5))=AV10
C
C          WRITE OUT OF COURSE RECORDS.
      WRITE(10'AV10) (CFILE(I),I=1,M)
```

```
C
C         CHECK FOR COURSE FILE OVERFLOW.
      IF(AV10.LE.MASLOK(1001))GO TO 900
      MASLOK(1001)= AV10
      IF(AV10.GT.OVER) GO TO 988
C
C         CHECK TO SEE IF THE NEXT COURSE IS TO BE ADDED, DELETED OR
C         CHANGED OR IF COURSE FILE ALTERATIONS ARE TERMINATED.
  900 IF(ADDTAG) GO TO 4
      IF(DELTAG .OR.CHATAG) RETURN 1
      IF(CHECK) GO TO 901
      ASSIGN 20 TO $IN
      GO TO 1
C
C         PRINT OUT OF NEW COURSE INFORMATION.
  800 IF(RTAG) WRITE(6,1108)CNAME
      IF(RTAG) WRITE(6,1109)(((RNAME(X*4-4+I),I=1,4),TEMFI1(X)),X=1,RC)
      IF(RTAG) CALL TWOCHK(TEMFI1)
      IF(TTAG) WRITE(6,1114)CNAME
      IF(TTAG) WRITE(6,1112)(((TNAME(X*12-12+I),I=1,12),TEMFI2(X)),
     1X=1,TC)
      IF(TTAG) CALL SORT(TEMFI2,TC)
      IF(TTAG) CALL DUPCHK(TEMFI2,TC)
      IF(STAG) WRITE(6,1115)CNAME
      IF(STAG) WRITE(6,1112)(((SNAME(X*12-12+I),I=1,12),TEMFI3(X)),
     1X=1,SC)
      IF(STAG) CALL SORT(TEMFI3,SC)
      IF(STAG) CALL DUPCHK(TEMFI3,SC)
      SC=SC+21
      MASLOK(CNAME(5))=AV10
C
C         WRITE OUT NEW COURSE RECORDS.
      WRITE(10'AV10)(TEMFIL(I),I=1,SC)
C
C         STORE LOCATION OF FIRST OPEN RECORD.
      MASLOK(1001)= AV10
C
C         CHECK FOR COURSE FILE OVERFLOW.
      IF(AV10.GT.OVER) GO TO 988
      GO TO 900
C
C         END OF COURSE FILE, ALTERATIONS EXIT.
  901 WRITE(7'1) MASLOK
  999 RETURN2
C
C         COURSE FILE OVERFLOW MESSAGE AND CARD FLUSH TO END OF COURSE
C         FILE ALTERATIONS.
  988 WRITE(6,1116) OVER
  977 READ(8,1000,END=901) CARD(1)
      GO TO 977
      END
```

CHAMOD

```
      SUBROUTINE CHAMOD(*,*)
         THIS ROUTINE MAKES CHANGES TO THE COURSE FILE.
      IMPLICIT INTEGER*2(A-Z), INTEGER*4($)
      COMMON/BLOK04/MASLOK(1025),CARD(17),AV10
      COMMON /BLOK06/ SNAME(1200),TNAME(120),RNAME(40),TEMFIL(121),
     1CFILE(2521)
      COMMON /BLOK14/ OVER
      LOGICAL *1 FINDER,CHECK,STAG,RTAG,TTAG,RERR,TERR,SERR,CONTAG,
     1ADDTAG,DELTAG,CHATAG
      DIMENSION TEMFI1(10), TEMFI2(10), TEMFI3(100)
      DIMENSION CNAME(5)
      DIMENSION  CFILE1(21)
      EQUIVALENCE (TEMFIL(1),TEMFI1(1)),(TEMFIL(11),TEMFI2(1))
      EQUIVALENCE( TEMFIL(22),TEMFI3(1))
      EQUIVALENCE (CFILE(1),CFILE1(1))
      DATA ADD,DEL,CHA/'AD','DE','CH'/
      DATA CORS,STUD,TECH,ROOM/'CO','ST','TE','RO'/
C
C
 1000 FORMAT(4A2,1X,12A2,T57,I4)
 1100 FORMAT('0',T10, 'COURSE FILE CHANGE ERROR '/,T10,
     1'THE CARD FOLLOWING AN ADD, DELETE, OR  CHANGE CARD MUST BE A COUR
     2SE CARD. ALL OTHERS WILL BE IGNORED.'/,T10,
     3'CARD READS AS FOLLOWS.'/T10,4A2,1X,12A2,22X,I4)
 1101 FORMAT('0',T10,'COURSE FILE CHANGE WARNING ',/,T10,4A2,1X,12A2,2X,
     1 I4,' WAS NOT ON RECORD. IT IS ASSUMED THAT ADD WAS INTENDED.')
 1103 FORMAT('0',T10, 'COURSE FILE CHANGE ERROR '/ ,T10,
     1'ILLEGAL CARD ENCOUNTERED.',2X,
     3'CARD READS AS FOLLOWS.'/T10,4A2,1X,12A2,22X,I4)
 1104 FORMAT('0',T10, 'COURSE FILE CHANGE ERROR '/ ,T10,
     1'TOO MANY STUDENTS WERE TO BE CHANGED. ALL OTHERS TO BE CHANGED IN
     2CLUDING THE FOLLOWING WILL BE INGNORED.'/ ,T10,
     3'CARD READS AS FOLLOWS.'/T10,4A2,1X,12A2,22X,I4)
 1105 FORMAT('0',T10, 'COURSE FILE CHANGE ERROR '/ ,T10,
     1'TOO MANY TEACHERS WERE TO BE CHANGED. ALL OTHERS TO BE CHANGED IN
     2CLUDING THE FOLLOWING WILL BE IGNORED.'/,T10,
     3'CARD READS AS FOLLOWS.'/T10,4A2,1X,12A2,22X,I4)
 1106 FORMAT('0',T10, 'COURSE FILE CHANGE ERROR '/ ,T10,
     1'TOO MANY ROOMS WERE TO BE CHANGED.  ALL OTHERS TO BE CHANGED INCL
     2UDING THE FOLLOWING WILL BE IGNORED.'/,T10,
     3'CARD READS AS FOLLOWS.'/T10,4A2,1X,12A2,22X,I4)
 1107 FORMAT('0',T10, 'COURSE FILE CHANGE ERROR '/ ,T10,
     1'TWO COURSE CARDS TOGETHER. THE FIRST CARD WILL BE IGNORED.' ,
     3'CARD READS AS FOLLOWS.'/T10,4A2,1X,12A2,22X,I4)
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 1108 FORMAT(//'0',T20,'COURSE', 4A2,2X,I4,'HAS BEEN CHANGED TO INCLUDE
     1THE FOLLOWING ROOMS, TEACHERS, STUDENTS.'///)
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 1109 FORMAT('0',T2,10(4A2,I4,1X))
 1110 FORMAT('0',/,4(2X,12A2,2X,I4))
 1111 FORMAT('0',T30,'COURSE FILE UPDATE HAS BEEN ABORTED BECAUSE THE
     1NUMBER OF RECORDS USED NOW EXCEEDS.',I5)
C
C
C         ****INITIALIZATION OF VARIABLES****
C
      ASSIGN 10 TO $IN
    1 DO 2 J=1,121
    2 TEMFIL(J)= 0
      CONTAG=.FALSE.
      CHATAG=.FALSE.
      ADDTAG=.FALSE.
      DELTAG=.FALSE.
      STAG=.FALSE.
      RTAG=.FALSE.
      TTAG=.FALSE.
      SERR=.FALSE.
      TERR=.FALSE.
      RERR=.FALSE.
      SC=0
      TC=0
      RC=0
      CHECK=.FALSE.
      GO TO $IN,(20,10)
C
C        STATEMENT 10 IS USED TO READ IN THE FIRST CARD FOLLOWING AN
C        ADD, DELETE OR CHANGE CARD.
 10   READ(8,1000,END=901) CARD
C
C        CHECK FOR COURSE CARD, ADD, DELETE OR CHANGE CARD AND TRANSFER
C        FOR FURTHER PROCESSING.
      IF(CARD(1).EQ.CORS) GO TO 20
      IF(CARD(1).EQ.CHA) GO TO 10
      IF(CARD(1).EQ.ADD) RETURN 1
      IF(CARD(1).EQ.DEL) RETURN 1
      IF(CONTAG) GO TO 10
C
C        ERROR--THE CARD FOLLOWING AN ADD, DELETE OR CHANGE CARD WAS
C        NOT A COURSE CARD, ADD, DELETE OR CHANGE CARD.
      WRITE (6,1100) CARD
      CONTAG=.TRUE.
      GO TO 10
C
C        STATEMENT 20 LOCATES THE RECORD OF THE COURSE TO BE PROCESSED.
   20 AV10=MASLOK(CARD(17))
C
C        STORE THE NAME AND NUMBER OF THE COURSE TO BE PROCESSED.
      DO 25 I= 1,4
   25 CNAME(I)= CARD(4+I)
      CNAME(5)= CARD(17)
C
C
      FINDER= AV10.EQ.0
      IF(FINDER) AV10= MASLOK(1001)
      FIND(10'AV10)
C
C        WARNING--THE COURSE NAMED DOES NOT EXIST.
C        FROM THIS POINT THE COURSE WILL BE ADDED RATHER THAN CHANGED.
C
      IF(FINDER) WRITE(6,1101) CARD
C
C        ****DETERMINE THE TYPE OF CARD AND TRANSFER FOR FURTHER
C        PROCESSING.****
C
  12  READ(8,1000, END=505) CARD
      IF(CARD(1).EQ.STUD) GO TO 200
      IF(CARD(1).EQ.TECH) GO TO 300
      IF(CARD(1).EQ.ROOM) GO TO 400
      IF(CARD(1).EQ.CORS) GO TO 500
      ADDTAG= CARD(1).EQ.ADD
      DELTAG=CARD(1).EQ.DEL
      CHATAG=CARD(1).EQ.CHA
      IF(ADDTAG.OR.CHATAG.OR.DELTAG) GO TO 500
C
C        ERROR--AN ILLEGAL CARD HAS BEEN ENCOUNTERED.
      WRITE(6,1103) CARD
      GO TO 12
C
C        ****STUDENT PROCESSING****
C
C        CHECK FOR STUDENT OVERFLOW.
  200 IF(SERR) GO TO 12
C        CHECK FOR STUDENTS ALL OR STUDENTS NONE CARD. (IGNORE EITHER
C        OF THESE CARDS)
      IF(CARD(17).EQ.9999.OR.CARD(17).EQ.0) GO TO 12
```

```
C       ****PRINT OUT THE NEW COURSE INFORMATION****
C
  501 WRITE(6,1108) CNAME
      IF(RTAG) WRITE(6,1109)(((RNAME(X*4-4+I),I=1,4),TEMFI1(X)),X=1,RC)
      IF(TTAG) WRITE(6,1110)(((TNAME(X*12-12+I),I=1,12 ),TEMFI2(X)),
     1 X=1,TC)
      IF(STAG) WRITE(6,1110)(((SNAME(X*12-12+I),I=1,12),TEMFI3(X)),
     1 X=1,SC)
      IF(RTAG) CALL TWOCHK(TEMFI1)
      IF(TTAG) CALL SORT(TEMFI2,TC)
      IF(TTAG) CALL DUPCHK(TEMFI2,TC)
      IF(STAG) CALL SORT(TEMFI3,SC)
      IF(STAG) CALL DUPCHK(TEMFI3,SC)
C
C       CHECK TO SEE IF THE COURSE BEING PROCESSED IS TO BE ADDED
C       OR CHANGED.
      IF(FINDER) GO TO 550
C
C       ****WRITE OUT THE NEW COURSE RECORD(S)****
C
      READ(10'AV10) CFILE1
      IF(CFILE1(21).GT.49) WRITE(10'AV10-1) TEMFIL
      IF(CFILE1(21).LE.49.AND.SC.LE.49) WRITE (10'AV10-1) (TEMFIL(I),I=1
     1,70)
      IF(CFILE1(21).LE.49.AND.SC.GT.49) GO TO 561
      IF (AV10.GT.OVER) GO TO 776
      GO TO 888
  561 MASLOK(CNAME(5))=MASLOK(1001)
      AV10=MASLOK(1001)
      WRITE(10'AV10) TEMFIL
      MASLOK(1001)=MASLOK(1001)+2
      IF (AV10.GT.OVER) GO TO 776
      GO TO 888
C
C       WRITE OUT THE ADD COURSE RECORD AND SET THE LOCATOR TO THE
C       FIRST OPEN RECORD.
  550 SC= SC+21
      MASLOK(CNAME(5))=MASLOK(1001)
      WRITE(10'AV10) (TEMFIL(J),J=1,SC)
      MASLOK(1001)=AV10
C
C       CHECK TO SEE IF A CHANGE, ADD OR DELETE CARD HAS BEEN
C       ENCOUNTERED. IF SO, TRANSFER.
C       IF A COURSE CARD OR THE END OF THE COURSE FILE ALTERATIONS
C       HAVE BEEN ENCOUNTERED, TRANSFER APPROPRIATELY.
  888 IF(CHATAG) GO TO 1
      IF(ADDTAG.OR.DELTAG) RETURN1
      IF(CHECK) GO TO 901
      ASSIGN 20 TO $IN
      GO TO 1
C
C       COURSE RECORD OVERFLOW WARNING MESSAGE IS PRINTED OUT AND
C       CARDS ARE FLUSHED TO THE END OF THE COURSE FILE ALTERATIONS.
  776 WRITE (6,1111) OVER
  777 READ(8,1000,END=901) CARD(1)
      GO TO 777
C
C       END OF COURSE FILE ALTERATIONS, EXIT.
  901 WRITE(7'1) MASLOK
  999 RETURN 2
      END
```

```
C
C         TAG SHOWING THAT STUDENTS HAVE BEEN ACCEPTED IS SET .TRUE.
      STAG =.TRUE.
C
C         KEEP A COUNT OF THE STUDENT CARDS ACCEPTED.
      SC=SC+1
C
C         CHECK FOR STUDENT CARD OVERFLOW. (LIMIT IS 100)
      IF(SC.EQ.101) GO TO 220
C
C         STORE TOTAL OF STUDENT CARDS ACCEPTED.
      TEMFIL(21)=SC
C
C         STORE STUDENT NAME AND NUMBER.
      TEMFI3(SC)= CARD(17)
      K=SC*12-16
      DO 202 I=5,16
  202 SNAME(K+I)= CARD(I)
      GO TO 12
C
C         STUDENT OVERFLOW MESSAGE IS POINTED OUT AND OVERFLOW TAG IS
C         SET TO .TRUE.
  220 SERR=.TRUE.
      SC=100
      WRITE(6,1104) CARD
      GO TO 12
C
C         ****TEACHER PROCESSING IS THE SAME FOR STUDENTS EXCEPT THE
C         OVERFLOW LIMIT IS 10 AND NO CHECK IS MADE FOR ALL OR NONE
C         CARDS****
C
  300 IF(TERR) GO TO 12
      TTAG=.TRUE.
      TC=TC+1
      IF(TC.GE.11) GO TO 320
      TEMFI2(TC)=CARD(17)
      K=TC*12-16
      DO 303 I=5,16
  303 TNAME(K+I)= CARD(I)
      GO TO 12
  320 TERR=.TRUE.
      TC=10
      WRITE (6,1105) CARD
      GO TO 12
C
C         ****ROOM PROCESSING IS THE SAME AS FOR TEACHERS****
C
  400 IF(RERR) GO TO 12
      RC=RC+1
      IF(RC.GE.11) GO TO 420
      RTAG=.TRUE.
      TEMFI1(RC)= CARD(17)
      K=RC*4-8
      DO 404 I=5,8
  404 RNAME(K+I) = CARD(I)
      GO TO 12
  420 RERR=.TRUE.
      RC=10
      WRITE(6,1106) CARD
      GO TO 12
C
C         CHECK FOR TWO COURSE CARDS TOGETHER. THE FIRST CARD IS IGNORED.
  500 IF(STAG.OR.TTAG.OR.RTAG) GO TO 501
      WRITE(6,1107) CARD
      GOTO 12
C
C         CHECK FOR THE END OF THE COURSE FILE ALTERATIONS.
  505 IF(.NOT.(STAG.OR.RTAG.OR.TTAG)) GO TO 901
C
C         THE END HAS BEEN ENCOUNTERED.
      CHECK=.TRUE.
C
```

```
      SUBROUTINE DELMOD (*,*)
C
C
C        THIS ROUTINE MAKES DELETIONS TO THE COURSE FILE.
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON/BLOK04/MASLOK(1025),CARD(17),AV10
      COMMON /BLOK06/SNAME(1200),TNAME(120),RNAME(40)
     1 ,TEMFIL(121),CFILE(2521)
      LOGICAL*1 CHECK,STAG,RTAG,TTAG,RERR,TERR,SERR
      LOGICAL*1 CONTAG,ADDTAG,DELTAG,CHATAG
      DIMENSION TEMFI1(10), TEMFI2(10), TEMFI3(100) ,CNAME(5)
      DIMENSION  CFILE1(10),CFILE2(10),CFILE3(2500)
      EQUIVALENCE (TEMFIL(1),TEMFI1(1)),(TEMFIL(11),TEMFI2(1))
      EQUIVALENCE( TEMFIL(22),TEMFI3(1))
      EQUIVALENCE (CFILE(1),CFILE1(1)),(CFILE(11),CFILE2(1))
      EQUIVALENCE (CFILE(22),CFILE3(1))
      DATA CORS,STUD,TECH,ROOM/'CO','ST','TE','RO'/
      DATA ADD,DEL,CHA/'AD','DE','CH'/
C
C
 1000 FORMAT(4A2,1X,12A2,T57,I4)
 1100 FORMAT('0',T10,'COURSEFILE DELETE ERROR '/,' ',
     1 T10,'THE CARD FOLLOWING AN ADD, DELETE, OR CHANGE CARD MUST',
     2 ' BE A COURSE CARD  ALL OTHERS WILL BE IGNORED.',/,' ',T10,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1101 FORMAT('0',T10,'COURSEFILE DELETE ERROR '/,' ',
     1 'THERE IS NO ',4A2,1X,12A2,20X,I4)
 1102 FORMAT('0',T10,'COURSEFILE DELETE ERROR '/,' ',
     1 T10,'ILLEGAL CARD ENCOUNTERED'/,T10,' ',
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1105 FORMAT('0',T10,'COURSEFILE DELETE ERROR '/,' ',
     1T10,'TOO MANY STUDENTS WERE TO BE DELETED. THIS AND FOLLOWING',
     2' CARDS TO BE DELETED WILL BE IGNORED.'/,' ',T10,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1106 FORMAT('0',T10,'COURSEFILE DELETE ERROR '/,' ',
     1 T10,'TOO MANY TEACHERS WERE TO BE DELETED. THIS AND FOLLOWING',
     2 ' CARDS WILL BE IGNORED.',/' ',T10,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1107 FORMAT('0',T10,'COURSEFILE DELETE ERROR '/,' ',
     1 T10,'TOO MANY ROOMS WERE TO BE DELETED. THIS AND FOLLOWING',
     2' CARDS TO BE DELETED WILL BE IGNORED.'/,' ',T10,
     3'CARD READS AS FOLLOWS.'/,' ',T10,4A2,1X,12A2,22X,I4)
 1108 FORMAT('0',T10,'THE FOLLOWING COURSE HAS BEEN DELETED.'/,' ',
     1 T10,'COURSE ',4A2,2X,I4)
 1109 FORMAT('0',T2,10(4A2,I4,1X))
 1110 FORMAT('0'/4(2X,12A2,2X,I4))
 1111 FORMAT ('0',T10, 'ALL STUDENTS HAVE BEEN DELETED FROM COURSE  ',
     2 4A2,2X,I4)
```

```
1112 FORMAT('0',T10,'THERE WERE NO STUDENTS IN COURSE',2X,4A2,2X,I4)
1113 FORMAT('0',T10,'THE FOLLOWING DELETIONS HAVE BEEN MADE TO ',
    1'COURSE  ',4A2,2X,I4)
1114 FORMAT('0',T10,'THE FOLLOWING ROOMS WERE DELETED')
1115 FORMAT('0',T10,'THE FOLLOWING TEACHERS WERE DELETED')
1116 FORMAT('0',T10,'THE FOLLOWING STUDENTS WERE DELETED')
C
C
C      ****INITIALIZATION OF VARIABLES****
C
   4 ASSIGN 10 TO $IN
  1  DO 2 I=1,121
  2  TEMFIL(I)=0
 3   CONTAG=.FALSE.
     CHECK=.FALSE.
     STAG=.FALSE.
     RTAG=.FALSE.
     TTAG=.FALSE.
     SERR=.FALSE.
     TERR=.FALSE.
     RERR=.FALSE.
     SC=0
     TC=0
     RC=0
     ADDTAG=.FALSE.
     CHATAG=.FALSE.
     DELTAG=.FALSE.
     GO TO $IN,(10,20)
C
C      STATEMENT 10 IS USED TO READ IN THE FIRST CARD FOLLOWING A
C    DELETE CARD.
 10  READ(8,1000,END=901) CARD
C
C       CHECK FOR ADD, DELETE, CHANGE OR COURSE CARD AND TRANSFER FOR
C       FURTHER PROCESSING.
C
     IF(CARD(1).EQ.CORS) GO TO 20
  11 IF(CARD(1).EQ.DEL) GO TO 10
     IF(CARD(1).EQ.ADD.OR.CARD(1).EQ.CHA) RETURN 1
     IF(CONTAG) GO TO 10
C
C      ERROR--CARD AFTER AN ADD, DELETE OR CHANGE CARD WAS NOT A
C       COURSE CARD.
     WRITE(6,1100) CARD
     CONTAG=.TRUE.
     GO TO 10
C
C      LOCATE THE RECORD(S) FOR THE COURSE BEING WORKED ON.
  20 AV10=MASLOK(CARD(17))
C
C      ERROR--COURSE DOES NOT EXIST.
     IF(AV10.EQ.0) WRITE(6,1101) CARD
     IF(AV10.EQ.0) GO TO 21
     FIND(10'AV10)
```

```
C
C        STORE COURSE NAME AND NUMBER.
      DO 25 I =1,4
   25 CNAME(I)= CARD(4+I)
      CNAME(5)= CARD(17)
C
C        STATEMENT 12 IS USED TO READ IN ALL CARDS FOLLOWING A COURSE
C        CARD.
   12 READ(8,1000,END=504) CARD
C
C        ****DETERMINE TYPE OF CARD AND TRANSFER FOR FURTHER PROCESSING*
C
      IF(CARD(1).EQ.STUD) GO TO 200
      IF(CARD(1).EQ.TECH) GOTO 300
      IF(CARD(1).EQ.ROOM) GO TO 400
      IF(CARD(1).EQ.CORS) GO TO 500
      ADDTAG= CARD(1).EQ.ADD
      DELTAG= CARD(1).EQ.DEL
      CHATAG= CARD(1).EQ.CHA
      IF(ADDTAG.OR.CHATAG.OR.DELTAG) GO TO  500
C
C        ERROR--ILLEGAL CARD ENCOUNTERED.
      WRITE(6,1102) CARD
      GO TO 12
C
C        AFTER ENCOUNTERING A CARD WITH A NON-EXISTENT COURSE NUMBER,
C        READ CARDS UNTIL ANOTHER COURSE, ADD, DELETE OR CHANGE CARD
C        HAS BEEN ENCOUNTERED OR THE END OF COURSE FILE ALTERATIONS
C        HAS BEEN ENCOUNTERED.
C
   21 READ(8,1000,END=901) CARD
      IF(CARD(1).EQ.CORS) GO TO 20
      IF(CARD(1).EQ.ADD.OR.CARD(1).EQ.DEL.OR.CARD(1).EQ.CHA) GO TO 11
      GO TO 21
C
C        ****PROCESS STUDENTS****
C
C        CHECK FOR STUDENT OVERFLOW.
  200 IF(SERR) GO TO 12
C
C        CHECK FOR STUDENTS ALL CARD.
      IF(CARD(17).EQ.9999) GO TO 215
C
C        CHECK FOR STUDENTS NONE CARD. (IGNORE THIS CARD)
      IF(CARD(17).EQ.0) GO TO 12
C
C        THE TAG SHOWING THAT STUDENT CARDS HAVE BEEN ACCEPTED IS SET
C        .TRUE.
      STAG=.TRUE.
C
C        KEEP A COUNT OF THE STUDENT CARDS ACCEPTED.
      SC=SC+1
  996 CONTINUE
```

```
C------------------------------------------------------------------
C        CHECK FOR STUDENT OVERFLOW. (LIMIT IS 100)
      IF(SC.EQ.101) GO TO 220
C------------------------------------------------------------------
C        STORE TOTAL OF STUDENT CARDS ACCEPTED.
      TEMFIL(21)= SC
C------------------------------------------------------------------
C        STORE STUDENT NUMBER AND NAME.
      TEMFI3(SC)= CARD(17)
      K=SC*12-16
      DO 202 I=5,16
  202 SNAME(K+I)= CARD(I)
      GO TO 12
C
C        ****STUDENTS ALL CARD HAS BEEN ENCOUNTERED****
C------------------------------------------------------------------
C        STORE ZERO STUDENTS FOR THE COURSE BEING WORKED ON.
  215 TEMFIL(21)=0
C------------------------------------------------------------------
C        ZERO OUT ANY NUMBERS THAT MAY ALREADY HAVE BEEN STORED.
      DO 216 I=1,100
  216 TEMFI3(I)=0
C
C        SET STUDENT OVERFLOW TAG TO .TRUE.
      SERR=.TRUE.
C
      SC=0
C        STUDENT COUNT IS SET TO ZERO.
C
C        THE TAG SHOWING THAT STUDENTS HAVE BEEN ACCEPTED IS SET .TRUE.
      STAG=.TRUE.
      WRITE(6,1111) CNAME
      GOTO 12
C------------------------------------------------------------------
C        STUDENT OVERFLOW ERROR MESSAGE IS PRINTED OUT AND OVERFLOW TAG
C        SET TO .TRUE.
  220 WRITE(6,1105) CARD
      SC=100
      SERR=.TRUE.
      GO TO 12
C
C        ****PROCESS TEACHERS****
C------------------------------------------------------------------
C        PROCESSING IS THE SAME FOR TEACHERS AS FOR STUDENTS EXCEPT THE
C        OVERFLOW LIMIT IS 10 AND NO CHECK IS MADE FOR ALL OR NONE
C        CARDS.
  300 IF(TERR) GO TO 12
      TTAG=.TRUE.
      TC=TC+1
      IF(TC.EQ.11) GO TO 220
      TEMFI2(TC)=CARD(17)
      K=TC*12-16
      DO 302 I=5,16
  302 TNAME(K+I)= CARD(I)
```

```
      GO TO 12
  320 TERR=.TRUE.
      TC=10
      WRITE(6,1106) CARD
      GO TO 12
C
C          ****PROCESS ROOMS****
C
C          PROCESSING IS THE SAME FOR ROOMS AS FOR TEACHERS.
  400 IF(RERR) GO TO 12
      RTAG=.TRUE.
      RC=RC+1
      IF(RC.GE.11) GO TO 420
      TEMFI1(RC)=CARD(17)
      K=RC*4-8
      DO 402 I=5,8
  402 RNAME(K+I)= CARD(I)
      GO TO 12
  420 RERR=.TRUE.
      WRITE(6,1107) CARD
      RC=10
      GO TO 12
C
C          CHECK FOR TWO COURSE CARDS TOGETHER.  THE COURSE NAMED ON THE
C          FIRST CARD IS COMPLETELY DELETED.
  500 IF(STAG.OR.RTAG.OR.TTAG) GO TO 505
C
C          DELETION OF A COMPLETE COURSE IS ACCOMPLISHED BY PLACING
C          A ZERO IN THE APPROPRIATE POSITION OF THE LOCATOR FILE.
      MASLOK(CNAME(5))=0
      WRITE(6,1108) CNAME
      GO TO 900
C
C         END OF COURSE FILE ALTERATIONS ENCOUNTERED SET CHECK .TRUE.
  504 CHECK=.TRUE.
C          STATEMENT 505 READS IN THE FIRST RECORD OF THE COURSE BEING
C          WORKED ON.
  505 READ(10'AV10) (CFILE(I),I=1,70)
      WRITE(6,1113) CNAME
C
C          CHECK--ARE ROOMS TO BE DELETED
      IF(.NOT.RTAG) GO TO 650
C
C          REPLACE THE NUMBERS OF THE ROOMS TO BE DELETED WITH A ZERO.
      DO 506 I=1,10
      IF(I.GT.RC) GO TO 507
      DO 506 J=1,10
  506 IF(TEMFI1(I).EQ.CFILE1(J)) CFILE1(J)=0
C
C          COMPACT THE ROOM LIST TO ELIMINATE THE ZEROS.
  507 P=10
      CALLSQUASH(CFILE1,P)
      WRITE(6,1114)
C
```

```
C         PRINT OUT THE ROOMS WHICH HAVE BEEN DELETED.
      WRITE(6,1109)(((RNAME(X*4-4+I) ,I=1,4),TEMFI1(X)),X=1,RC)
C
C         ****TEACHER DELETIONS ARE MADE THE SAME WAY AS ROOM
C         DELETIONS****
C
  650 IF(.NOT.TTAG) GO TO 670
      DO 676 I=1,10
      IF(I.GT.TC) GO TO 677
      DO 676 J=1,10
  676  IF(TEMFI2(I).EQ.CFILE(J)) CFILE2(J)=0
  677 P=10
      CALLSQUASH(CFILE2,P)
      WRITE(6,1115)
      WRITE(6,1110)(((TNAME(X*12-12+I),I=1,12),TEMFI2(X)),X=1,TC)
C
C          ****DELETION OF STUDENTS****
C
  670 IF(.NOT.STAG) GO TO 701
C
C  CHECK FOR DELETE ALL
      IF(TEMFIL(21).EQ.0) CFILE(21)=0
      IF(TEMFIL(21).EQ.0) GO TO 701
C          WARNING--NO STUDENTS WERE IN THE COURSE BEING WORKED ON.
      IF(CFILE(21).EQ.0) WRITE(6,1112) CNAME
      IF(CFILE(21).EQ.0) GO TO 750
C
C          READ IN ALL STUDENTS IN THE COURSE.
      IF(CFILE(21).GT.49) P= CFILE(21)-49
      IF(CFILE(21).GT.49) READ(10'AV10)(CFILE3(49+I),I=1,P)
      P=CFILE(21)
C
C          DELETION OF STUDENTS IS ACCOMPLISHED BY THE SAME METHOD
C          AS FOR TEACHERS AND ROOMS.
C
      DO 680 I=1,SC
      DO681 J=1,P
      IF(TEMFI3(I).EQ.CFILE3(J)) GO TO 682
  681 CONTINUE
      GO TO 680
  682 CFILE3(J)= 0
  680 CONTINUE
      CALL SQUASH(CFILE3,P)
      CFILE(21)=P
C
C          PRINT OUT THE STUDENTS WHO HAVE BEEN DELETED.
      WRITE(6,1116)
      WRITE(6,1110)(((SNAME(X*12-12+I),I=1,12),TEMFI3(X)),X=1,SC)
C
C          WRITE OUT THE ALTERED COURSE RECORDS.
C
```

```
  700 IF(STAG) GO TO 750
  701 AV10=AV10-1
      WRITE(10'AV10)(CFILE(I),I=1,70)
      GO TO 900
  750 AV10= MASLOK(CNAME(5))
      P=CFILE(21)+21
C
      WRITE(10'AV10)(CFILE(I),I=1,P)
C         IF AN ADD, DELETE OR CHANGE CARD OR COURSE CARD, OR THE END
C         TO THE APPROPRIATE PLACE.
C         OF THE COURSE FILE ALTERATIONS HAS BEEN ENCOUNTERED, TRANSFER
C
  900 IF(DELTAG) GO TO 4
      IF (ADDTAG.OR.CHATAG) RETURN 1
      IF(CHECK) GO TO 901
      ASSIGN 20 TO SIN
      GO TO 1
  901 WRITE(7'1) MASLOK
C
C         END OF COURSE FILE ALTERATIONS, EXIT.
  999 RETURN 2
      END
```

NAMCON

```
      SUBROUTINE NAMCON (F1,F2,F3,F4,XY,F5,FN,F7)
C
C
C        THE PURPOSE OF THIS SUBROUTINE IS TO EFFECT CHANGES IN THE
C        COURSE NAME FILE, THE ROOM NAME FILE, STUDENT NAME FILE,
C        TEACHER NAME FILE IN THAT ORDER.
C
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON/BLOK04/MASLOK(1025),CARD(17),AV10
      LOGICAL*1 CONTAG,CTAG,CACERR,CDCERR,CCCERR,ADDIT,DELIT,CHAIT,
     1 YES,YES2
      INTEGER*4 FN
      DIMENSION CNAMET(2600),CNAMED(1300)
      DIMENSION NAME(4), NAME2(12)
      DIMENSION NNAME(12)
      DIMENSION NFILE(500), NFILE2(564)
      EQUIVALENCE (NFILE,NFILE2),(NAME,NAME2)
      DATA NNAME/'NO NAME                  '/
      DATA CORS,STUD,TECH,ROOM/'CO','ST','TE','RO'/
      DATA ADD,DEL,CHA/'AD','DE','CH'/
C
C
 1000 FORMAT(4A2,1X,12A2,T57,I4)
 1100 FORMAT('0',T10,'COURSE NAME CHANGES ERROR '/,' ',
     1 T10,'ONLY COURSE CARDS ALLOWED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1101 FORMAT('0',T10,'ROOM   NAME CHANGES ERROR '/,' ',
     1 T10,'ONLY ROOM CARDS ALLOWED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1102 FORMAT('0',T10,'STUDENT NAME CHANGES ERROR '/,' ',
     1 T10,'ONLY STUDENT CARDS ALLOWED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1103 FORMAT('0',T10,'TEACHER NAME CHANGES ERROR '/,' ',
     1 T10,'ONLY TEACHER CARDS ALLOWED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1104 FORMAT('0',T10,'COURSE NAME CHANGES ERROR '/,' ',
     1 T10,'FIRST CARD MUST BE  AN ADD, OR CHANGE, OR DELETE CARD.'
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1105 FORMAT('0',T10,'ROOM NAME CHANGES ERROR '/,' ',
     1 T10,'FIRST CARD MUST BE  AN ADD, OR CHANGE, OR DELETE CARD.'
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1106 FORMAT('0',T10,'STUDENT NAME CHANGES ERROR'/,' ',
     1 T10,'FIRST CARD MUST BE  AN ADD, OR CHANGE, OR DELETE CARD.'
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1107 FORMAT('0',T10,'TEACHER NAME CHANGES ERROR '/,' ',
     1 T10,'FIRST CARD MUST BE  AN ADD, OR CHANGE, OR DELETE CARD.'
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1108 FORMAT('0',T10,'TOO MANY COURSE NAMES WERE TO BE ADDED. THE FOLLOW~~
     1ING CARD AND OTHERS TO BE ADDED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1109 FORMAT('0',T10,'TOO MANY ROOM NAMES WERE TO BE ADDED. THE FOLLOWI~~
     1G CARD AND OTHERS TO BE ADDED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1110 FORMAT('0',T10,'TOO MANY STUDENT NAMES WERE TO BE ADDED. THE FOLLO~~
     1WING CARD AND OTHERS TO BE ADDED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1111 FORMAT('0',T10,'TOO MANY TEACHER NAMES WERE TO BE ADDED. THE FOLLO~~
     1WING CARD AND OTHERS TO BE ADDED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
 1112 FORMAT('0',T10,'TOO MANY COURSE NAMES WERE TO BE CHANGED. THE FOLL
     1OWING CARD AND OTHERS TO BE ADDED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1113 FORMAT('0',T10,' TOO MANY ROOM NAMES WERE TO BE CHANGED. THE FOLLO~~
     1WING CARD AND OTHERS TO BE CHANGED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1114 FORMAT('0',T10,'TOO MANY STUDENT NAMES WERE TO BE CHANGED. THE FOL~~
     1LOWING CARD AND OTHERS TO BE CHANGED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1115 FORMAT('0',T10,'TOO MANY TEACHER NAMES WERE TO BE CHANGED. THE FOL~~
     1LOWING CARD AND OTHERS TO BE CHANGED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1116 FORMAT('0',T10,'TOO MANY COURSE NAMES WERE TO BE DELETED. THE  FOL~~
     1LOWING CARD AN OTHERS TO BE DELETED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1117 FORMAT('0',T10,'TOO MANY ROOM NAMES WERE TO BE DELETED. THE FOLLOW~~
     1ING CARD AN OTHERS TO BE DELETED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1118 FORMAT('0',T10,'TOO MANY STUDENT NAMES WERE TO BE DELETED. THE FOL~~
     1LOWING CARD AN OTHERS TO BE DELETED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1119 FORMAT('0',T10,'TOO MANY TEACHER NAMES WERE TO BE DELETED. THE FOL~~
     1LOWING CARD AN OTHERS TO BE DELETED WILL BE IGNORED.',
     2 'CARD READS AS FOLLOWS.'//,' ',T10,4A2,1X,12A2,22X,I4)
~~1120 FORMAT('1',T32,'THE FOLLOWING ALTERATIONS HAVE BEEN MADE TO THE CO~~
     1URSE NAME FILE')
 1121 FORMAT('1',T32,'THE FOLLOWING ALTERATIONS HAVE BEEN MADE TO THE RO
~~     1OM NAME FILE')~~
 1122 FORMAT('1',T32,'THE FOLLOWING ALTERATIONS HAVE BEEN MADE TO THE ST
     1UDENT NAMEFILE')
~~1123 FORMAT('1',T30,'THE FOLLOWING ALTERATIONS HAVE BEEN MADE TO THE TE~~
     1ACHER NAME FILE')
 1124 FORMAT(' ',T5,'ADDED OR CHANGED ',12A2,I4,2X,'IS NOW ',12A2,I4)
~~1125 FORMAT(' ',T5,'DELETED ',12A2,2X,I4)~~
 1126 FORMAT('0',///)
 1127 FORMAT(' ',T5,'ADDED OR CHANGED ',4A2,I4,2X,'IS NOW ',4A2,I4)
~~1128 FORMAT(' ',T5,'DELETED ',4A2,2X,I4)~~
      DEFINE FILE  1(8,250,U,AV)
      DEFINE FILE 2(4,250,U,AV)
~~      DEFINE FILE 3(54,282,U,AV)~~
      DEFINE FILE 4(11,282,U,AV)
C
```

```
C
C       STATEMENTS  0 THRU 9 ARE INITIALIZATION STATEMENTS
C
  999 CONTINUE
      DO 1 I=13,1300,13
      CNAMET(I)= 0
      CNAMET(I+1300)= 0
    1 CNAMED(I)= 0
      DO 2 I=5,1300,5
      CNAMET(I)= 0
      CNAMET(I+1300)= 0
    2 CNAMED(I)= 0
      SAV= XY-1
      ASSIGN 104 TO SCON
      CONTAG=.FALSE.
      CTAG=.FALSE.
      CACERR=.FALSE.
      CCCERR= .FALSE.
      CDCERR=.FALSE.
      CAC=0
      CAT=0
      CCC=0
      CDC=0
      A=1
      B=1
      AV=1
      GO TO 102
C
C       IF THE FIRST CARD IS NOT AN ADD, DELETE, OR CHANGE CARD,
C       STATEMENT 99 IS USED TO READ IN THE CARDS UNTIL SUCH A CARD
C       IS FOUND.
C
   99 READ(8,1000,END=200) CARD
      GO TO 102
C
C       STATEMENT 101 READS IN CARDS UNDER NORMAL CONDITIONS.
C
  101 READ(8,1000,END=201) CARD
C
C       STATEMENTS 102 THRU 102+2 CHECK FOR ADD,DELETE, CHANGE CARDS
C
  102 ADDIT=CARD(1).EQ.ADD
      DELIT=CARD(1).EQ.DEL
      CHAIT=CARD(1).EQ.CHA
C
C       STATEMENT 300-1 CONTROLS WHICH FILE WORK IS BEING DONE ON.
C       STATEMENTS 300 THRU 303 CHECK ON NAME CARDS ACCORDING TO THE
C       FILE BEING WORKED ON.
```

```
C
      GO TO   (300,301,302,303),F1
  300 IF(CARD(1).EQ.CORS) GO TO $CON,(121,130,160,104)
      GO TO 202
  301 IF(CARD(1).EQ.ROOM) GO TO $CON,(121,130,160,104)
      GO TO 202
  302 IF(CARD(1).EQ.STUD) GO TO $CON,(121,130,160,104)
      GO TO 202
  303 IF(CARD(1).EQ.TECH) GO TO $CON,(121,130,160,104)
C
C         STATEMENT 202 IS A CONTROL CARD CHECK.
C         STATEMENTS 202+1 THRU 202+4  ERROR MESSAGE WRITEOUT.
C
  202 IF(ADDIT.OR.CHAIT.OR.DELIT) GO TO 120
      IF(CTAG.AND.F1.EQ.1) WRITE(6,1100) CARD
      IF(CTAG.AND.F1.EQ.2) WRITE(6,1101) CARD
      IF(CTAG.AND.F1.EQ.3) WRITE(6,1102) CARD
      IF(CTAG.AND.F1.EQ.4) WRITE (6,1103) CARD
      IF(CTAG) GO TO 101
C
C         STATEMENTS 104 THRU 120-1 MAKE CERTAIN THAT THE FIRST CARD
C         IS A CONTROL CARD.
C
  104 IF(CONTAG) GO TO 99
      IF(.NOT.CONTAG.AND.F1.EQ.1) WRITE(6,1104) CARD
      IF(.NOT.CONTAG.AND.F1.EQ.2) WRITE(6,1105) CARD
      IF(.NOT.CONTAG.AND.F1.EQ.3) WRITE(6,1106) CARD
      IF(.NOT.CONTAG.AND.F1.EQ.4) WRITE(6,1107) CARD
      CONTAG=.TRUE.
      GO TO 99
C
C         STATEMENTS 120 THRU 120+2 DECIDE IF THE NAME CARD IS TO BE
C         ADDED, DELETED, OR CHANGED.
C
  120 IF(ADDIT) ASSIGN 121 TO $CON
      IF(DELIT) ASSIGN 160 TO $CON
      IF(CHAIT) ASSIGN 130 TO $CON
      GO TO  101
C
C         STATEMENT 121 CHECKS FOR STUDENT OVERFLOW, STUDENTS ALL OR
C         STUDENTS NONE.
C         STATEMENTS 121 THRU 124+1 LOAD IN NAMES AND NUMBERS TO
C         BE ADDED.
C
  121 IF(CACERR.OR.CARD(17).EQ.9999.OR.CARD(17).EQ.0) GO TO 101
      CAC= CAC+1
      CAT= CAT+1
      IF(CAC.GE.101) GO TO 125
  123 CTAG=.TRUE.
      CATT=CAT*F2-F3
      DO 124 I= 5,F4
  124 CNAMET(CATT+I)= CARD(I)
      CNAMET(CATT+F3)= CARD(17)
      GO TO 101
  125 CACERR=.TRUE.
      CAC=100
      IF(F1.EQ.1) WRITE(6,1108) CARD
      IF(F1.EQ.2) WRITE(6,1109) CARD
      IF(F1.EQ.3) WRITE(6,1110) CARD
      IF(FI.EQ.4) WRITE(6,1111) CARD
      GO TO 101
C
C        STATEMENTS 130 THRU 160-2 LOAD IN NAMES AND NUMBERS TO BE
C        CHANGED.
C
  130 IF(CCCERR.OR.CARD(17).EQ.9999.OR.CARD(17).EQ.0) GO TO 101
      CCC=CCC+1
      CAT=CAT+1
      IF(CCC.GE.101) GO TO 135
      GO TO 123
  135 CCCERR=.TRUE.
      CCC=100
      IF(F1.EQ.1) WRITE(6,1112) CARD
      IF(F1.EQ.2) WRITE(6,1113) CARD
      IF(F1.EQ.3) WRITE(6,1114) CARD
      IF(F1.EQ.4) WRITE(6,1115) CARD
      GO TO 101
C
C        STATEMENTS 160 THRU 201-1 LOAD IN NAMES AND NUMBERS TO BE
C        DELETED.
C
  160 IF(CDCERR.OR.CARD(17).EQ.9999.OR.CARD(17).EQ.0) GO TO 101
      CDC=CDC+1
      IF(CDC.GE.101) GO TO 165
      CTAG=.TRUE.
      CDT=CDC*F2-F3
      DO 164 I=5,F4
  164 CNAMED(CDT+I)= CARD(I)
      CNAMED(CDT+F3) = CARD(17)
      GO TO 101
  165 CDCERR=.TRUE.
      CDC=100
      IF(F1.EQ.1) WRITE(6,1116) CARD
      IF(F1.EQ.2) WRITE(6,1117) CARD
      IF(F1.EQ.3) WRITE (6,1118) CARD
      IF(F1.EQ.4) WRITE(6,1119) CARD
      GO TO 101
C
C        STATEMENTS 201 THRU 201+4 WRITE OUT HEADINGS.
C
  201 IF(F1.EQ.1) WRITE(6,1120)
      IF(F1.EQ.2) WRITE(6,1121)
      IF(F1.EQ.3) WRITE(6,1122)
      IF(F1.EQ.4) WRITE(6,1123)
      WRITE(6,1126)
C
C        STATEMENTS 170-2 AND 170-1 ORDER THE NAMES.
```

```
C
      IF(CAT.NE.0)CALL BLKSRT(CNAMET,CAT,F2)
      IF(CDC.NE.0) CALL BLKSRT(CNAMED,CDC,F2)
C
C         STATEMENTS 170 AND 170+1 DECIDE IF CHANGES AND ADDITIONS,
C         OR DELETIONS ARE TO BE MADE.
C
  170 YES=(CNAMET(A*F2).NE.0.AND.CNAMET(A*F2).LT.XY).OR.
     1(CNAMED(B*F2).NE.0.AND.CNAMED(B*F2).LT.XY)
      IF(.NOT.YES) GO TO 180
      FIND(FN'AV)
C
C         STATEMENT 170+3 DECIDES IF ONLY CHANGES AND/OR ADDITIONS
C         ARE TO BE MADE.
C
      YES2= CNAMET(A*F2).NE.0.AND.CNAMET(A*F2).LT.XY
C
C         STATEMENTS 170+4 THRU 190 MAKE ADDITIONS AND/OR CHANGES TO
C         THE NAME FILE RECORD PRESENTLY IN CORE.
C
      IF(F5.EQ.4) READ(FN'AV) NFILE
      IF(F5.EQ.12) READ(FN'AV) NFILE2
      IF(.NOT.YES2) GO TO 193
      DO 177 X=A,CAT
      IF(CNAMET(X*F2).GT.XY) GO TO 190
      K= (CNAMET(X*F2)-(AV-2)*SAV)*F5-F5
      Z=X*F2-F2
      DO 166 I=1,F5
      NAME(I)=NFILE(K+I)
  166 NFILE(K+I)= CNAMET(Z+I)
      IF(F5.EQ.4) WRITE(6,1127) NAME,CNAMET(X*F2),(CNAMET(Z+I),I=1,5)
  177 IF(F5.EQ.12) WRITE(6,1124) NAME2,CNAMET(X*F2),(CNAMET(Z+I),I=1,13)
      CNAMET(A*F2)= 0
      GOTO 191
  190 A=X
C
C         STATEMENT 191 CHECKS TO SEE IF NAMES ARE TO BE CHANGED.
C
  191 IF(CNAMED(B*F2).NE.0.AND.CNAMED(B*F2).LT.XY) GO TO 193
      GO TO 180
C
C         STATEMENTS 193 THRU 192 MAKE DELETIONS TO THE NAME FILE
C         RECORD PRESENTLY IN CORE.
C
  193  DO 155 X=B,CDC
      IF(CNAMED(X*F2).GT.XY) GO TO 192
      K= (CNAMED(X*F2)-(AV-2)*SAV)*F5-F5
      Z=X*F2-F2
      DO 344 I=1,F5
  344 NFILE(K+I)= NNAME(I)
      IF(F5.EQ.4) WRITE(6,1128)(CNAMED(Z+I),I=1,5)
  155 IF(F5.EQ.12)WRITE(6,1125)(CNAMED(Z+I),I=1,13)
      CNAMED(B*F2)= 0
      GO TO 180
  192 B=X
```

```
C
C       STATEMENTS 180 THRU 180+2 WRITE OUT THE ALTERED NAME FILE
C       RECORD.
C       STATEMENT 180+3 IF NECESSARY UPDATES THE ASSOCIATED VARIABLE.
C       STATEMENT 180+4 CHECKS TO SEE IF ALTERATIONS ON THE NAME FILE
C       BEING WORKED ON ARE COMPLETED.
C       STATEMENT 180+5 UPDATES THE XY PARAMETER.
C
  180 IF(YES) AV=AV-1
      IF(YES.AND.F5.EQ.4) WRITE(FN'AV) NFILE
      IF(YES.AND.F5.EQ.12) WRITE(FN'AV) NFILE2
      IF(.NOT.YES)AV=AV+1
      IF(AV.GT.F7) RETURN
      XY=XY+SAV
      IF(CNAMET(A*F2).EQ.0.AND.CNAMED(B*F2).EQ.0) RETURN
      GO TO 170
  200 RETURN
      END
      BLOCK DATA
      IMPLICIT INTEGER*2(A-Z), INTEGER*4($)
      COMMON /BLOK14/ OVER
      DATA OVER/1050/
      END
```

TWOCHK
AND
NTRY SQUASH

```
      SUBROUTINE TWOCHK(ROOM)
C
C
C        THIS  SUBROUTINE IS USED TO CHECK FOR AND ELIMINATE DUPLICATIONS
C        IN ROOM LISTS.
C     'ROOM' IS THE NAME OF A SINGLE DIMENSIONED ARRAY.
C     'TAG' IS USED  AS A CHECK TO SEE IF THERE WERE ANY DUPLICATIONS.
C        STATEMENTS 0 THRU 20+1 LOCATE DUPLICATIONS AND REPLACE THEM
C        WITH  ZEROS.  STATEMENTS 20+1 THRU 30 SHIFT APPROPRIATE PARTS OF
C        THE ARRAY TO ELIMINATE THE ZEROS.  STATEMENTS 30+1 THRU 40
C        ZERO OUT THE VACATED END OF THE ARRAY.
C
C     THE ENTRY POINT SQUASH IS USED BY SUBROUTINE DELMOD TO COMPACT
C     VARIOUS ARRAYS AFTER DELETIONS HAVE BEEN MADE BY THAT ROUTINE.
C     J IS SET EQUAL TO THE NEW NUMBER OF ITEMS IN THE ARRAY.
C
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      LOGICAL*1 TAG
      DIMENSION ROOM(1)
C
C
      J=10
      TAG=.TRUE.
      A=0
   10 A=A+1
      DO 20I= A,9
      IF(ROOM(A).NE.ROOM(I+1)) GO TO 20
      ROOM(I+1)=0
      TAG=.FALSE.
   20 CONTINUE
      IF(A.NE.9) GO TO 10
      IF(TAG) RETURN
C
      ENTRY SQUASH (ROOM,J)
      K=0
      DO 30 I=1,J
      ROOM(I-K)= ROOM(I)
      IF(ROOM(I).EQ.0)K=K+1
   30 CONTINUE
      K=J-K+1
      DO 40 I= K,J
   40 ROOM(I)=0
      J=K -1
      RETURN
      END
```

BLKSRT

```
      SUBROUTINE BLKSRT(A,B,C)
C
C     THIS ROUTINE IS USED TO SORT COURSE,TEACHER,AND STUDENT NAMES PRIOR
C     TO THEIR USE IN MAKEING CHANGES TO THESE NAME FILES.
C     'A' IS THE NAME OF A SINGLE DIMENSIONED ARRAY
C     'B' IS THE NUMBER OF ITEMS TO BE SORTED
C     'C' IS THE SIZE OF THE BLOCKS MEASURED IN HALF WORDS
C        NOTE  THE LAST HALF WORD MUST CONTAIN AN INTEGER NUMBER.
C        THE SORT IS MADE ON THIS NUMBER.
C
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      DIMENSION A(1)
C
C
      IF(B.LT.2) RETURN
      N=(B-1)*C
      DO 10 J=C,N,C
      I=J
    9 IF(A(I).LE.A(I+C)) GO TO 10
      L=I-C+1
      DO 11 K=L,I
      TEMP=A(K)
      A(K)=A(K+C)
   11 A(K+C)=TEMP
      I=I-C
      IF(I)10,10,9
   10 CONTINUE
      RETURN
      END
```

CONDENSE

```
C       MAIN ROUTINE CONDNZ
C
C             THIS ROUTINE PRODUCES A LIST OF ALL STUDENT NUMBERS IN USE,
C       AND THE RELATIVE LOCATION IN THE STUDENT-SCHEDULE FILE WHERE
C       STUDENT DATA MAY BE FOUND.  SEE PNC 4101
C
C
        IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
        INTEGER*4 NONE/'NO N'/, STDNT(282), M*2/0/,ENDD*2/47/,INDEX*2/0/
        DIMENSION LIST(2500)
C
C
        DEFINE FILE 03 (54,282,U,AV03)
        DEFINE FILE 17(8,625,U,AV17)
C
C       READ DATA SET 3.  THERE ARE 54 RECORDS IN DATA SET 3.
C       EACH RECORD EXCEPT THE LAST HAS 47 ENTRIES (NAMES).
C
        DO 100 I= 1,54
        IF (I.EQ.54) ENDD = 9
C                                      THERE ARE ONLY 9 ENTRIES IN RECORD 54.
        READ (03'I) STDNT
C       GO THROUGH ARRAY IN 6 WORD (24 BYTE) STEPS.  IF 'NO NAME' FOUND,
C       IGNORE ENTRY, PUT ZERO IN STUDENT-NUMBER POSITION OF LIST.
C       OTHERWISE, PUT NEXT INTEGER VALUE IN THE STUDENT NUMBER POSITION
C       OF THE LIST.  START WITH INTEGER VALUE 1.
C
        K = -5
        DO 100 L=1, ENDD
        INDEX = INDEX + 1
        K = K + 6
        IF (STDNT(K) .EQ. NONE) GO TO 90
        M = M + 1
        LIST(INDEX) = M
        GO TO 100
   90   LIST(INDEX) = 0
  100   CONTINUE
C
C       WRITE OUT LIST AS RECORDS 7 + 8.  EACH RECORD IS 2500 BYTES LONG
C
C
        WRITE (17'7) LIST
        CALL ZEROIT
        END
```

ZEROIT

```
      SUBROUTINE ZEROIT
      INTEGER * 4 A(896)/896*0/,AV25/1/
      DEFINE FILE 25(40,896,U,AV25)
      DO 40 I=1,40
   40 WRITE(25'AV25) A
      CALL EXIT
      END
```

```
C     THIS ROUTINE READS THE REQUEST FILE AND CREATES TWO NEW FILES,
C     DATA SET 16 (AN ABBREVIATED REQUEST FILE) AND DATA SET 17 (A
C     PRIORITY FILE). IT HAS CAPACITY FOR 2500 REQUESTS WITH A MAXIMUM
C     OF 2500 STUDENTS IN EACH REQUEST. EACH ABBREVIATED REQUEST
C     CONTAINS THE FOLLOWING INFORMATION (MARGINAL NUMBER = BYTES USED)-
C
C          (20)  1--THE CODE NUMBERS FOR THE 10 ROOMS.
C          (20)  2--THE CODE NUMBERS FOR THE 10 TEACHERS.
C           (2)  3--THE NUMBER OF PERIODS REQUESTED.
C           (2)  4--THE MAXIMUM NUMBER OF STUDENTS THAT THE TEACHER WILL
C                   ALLOW TO BE ABSENT BEFORE CANCELLING THE REQUEST.
C                   UNLESS SPECIFIED HERE, IT IS ASSUMED TO BE ONE THIRD
C                   OF THE CLASS ENROLLMENT.
C           (2)  5--THE 'SHORT' CODE NUMBER WHICH INDICATES ONE OF THE
C                   FOLLOWING OPTIONS--
C
C                          0--THE PERIODS MAY NOT BE SHORTENED.
C                             THE STUDENT MAY MISS NO PERIODS.
C                          1--THE PERIODS MAY BE SHORTENED.
C                             THE STUDENT MAY MISS NO PERIODS.
C                          2--THE PERIODS MAY NOT BE SHORTENED.
C                             THE STUDENT MAY MISS SOME PERIODS.
C                          3--THE PERIODS MAY BE SHORTENED.
C                             THE STUDENT MAY MISS SOME PERIODS.
C
C           (2)  6--THE PERIOD AT WHICH THE COURSE MUST START (IF NOT
C                   SPECIFIED HERE, THE COMPUTER WILL CHOOSE ONE).
C           (4)  7--THE PERIODS BETWEEN WHICH THE COURSE MUST BE
C                   STARTED.
C           (2)  8--THE NUMBER OF GROUPS IN A GROUP REQUEST OR THE NUMBER
C                   OF PERIODS IN AN OPEN REQUEST.
C           (2)  9--THE CODE NUMBER OF THE TYPE OF REQUEST.
C
C                          0--REGULAR
C                          1--NO PRIORITY
C                          2--GROUP
C                          3--MAY
C                          4--OPEN
C                          5--OPEN MAY
C
C           (2) 10--THE NUMBER OF STUDENTS IN THE COURSE.
C           (2) 11--THE CODE NUMBER FOR THE COURSE.
C           (2) 12--TWO BLANK BYTES RESERVED FOR FUTURE USE.
C
C     THE PRIORITY FILE IS A POSITION SENSITIVE SET OF TWO BYTE
C     INTEGERS. EACH INTEGER CONTAINS THE RELATIVE RECORD NUMBER OF ONE
C     OF THE ABBREVIATED REQUESTS. ITS POSITION IN THE FILE INDICATES
C     ITS PRIORITY, THOSE AT THE BEGINNING OF THE FILE HAVING HIGHEST
```

```
C     PRIORITY. THIS FILE ALSO CONTAINS A POSITION SENSITIVE SET OF 500
C     TWO BYTE INTEGERS-THE POSITION INDICATING THE TEACHER CODE NUMBER
C     AND THE VALUE OF THE INTEGER BEING THE TOTAL NUMBER OF PERIODS
C     REQUESTED BY THE TEACHER IN ALL BUT 'NO-PRIORITY' REQUESTS.
C
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      INTEGER*4 STUDEN/4HUDEN/,PERIOD/4HRIOD/,COURSE/4HURSE/,
     1TEACH/4HACHE/,ROOM/4HOM  /,GROUP/4HOUPS/,MAYSCH/4HY SC/,
     2OPEN/4HEN  /,OPMAY/4HEN (/,BETPER/4HTWEE/,ANDPER/4HD PE/,
     3START/4HARTI/,CONFI/4HNFLI/,PRIOR/4HIORI/,NOPRI/4H PRI/,
     4NOSHOR/4H SHO/,NOISHO/4H IND/,TIME01,TIME02
      LOGICAL*1 STUREQ
C
C     THE ABOVE VARIABLES HAVE THE UNUSUAL VALUES ASSIGNED THEM BECAUSE
C     THE PROGRAM IS DESIGNED TO CHECK THE THIRD THROUGH THE SIXTH
C     CHARACTERS OF THE TYPE NAME ON EACH INPUT CARD.
C
      LOGICAL*1 FLAG/F/,STUFIL/F/,NOSTUD/F/,GRP/F/,GRPMAY/F/,OP/F/,
     1OPEMAY/F/,ROMS
C
      DIMENSION M(1000),TEMPST(2528),ROOMS(10),TECHER(10),CARD(9),
     1FILLER(28),TEACHR(500),RMS(10),TCHS(10),TEMSTD(2528),TEMFIL(3,2500
     2)
C
C     M IS A POSITION SENSITIVE ARRAY WHICH HOLDS DATA SET 15 IN CORE.
C     EACH WORD IN THE ARRAY CONTAINS THE RELATIVE RECORD NUMBER OF A
C     RECORD WHICH BEGINS THE INFORMATION ON ONE OF THE 1000 POSSIBLE
C     COURSES. ITS POSITION IN THE ARRAY IS ALSO THE COURSE CODE
C     NUMBER. M(5) WOULD THEREFORE BE THE RELATIVE RECORD NUMBER OF THE
C     RECORD ON WHICH BEGINS THE INFORMATION ON COURSE NUMBER FIVE.
C
      DATA FILLER/28*0/,BLANK/0/,VA16/1/,VA17/1/,TEACHR/500*2/,NOM/0/,
     $ DUMDUM/3/
      INTEGER*4 CARD
      DATA AV7,AV10/1,1/
      DEFINE FILE 7(1,2050,L,AV7)
      DEFINE FILE 10(1100,35,U,AV10)
      DEFINE FILE 16(600,162,L,VA16)
      DEFINE FILE 17 (8,625,U,VA17)
C
C
 1000 FORMAT(A2,A4,3X,6A4,24X,I3)
 2000 FORMAT(1H0,'INVALID FIRST CARD ENCOUNTERED.',/1H ,'CARD READS AS F
     1OLLOWS--',5X,A2,A4,3X,6A4,23X,I4)
 3000 FORMAT(1H0,'INVALID CARD ENCOUNTERED--COURSE ',I5,/1H ,'CARD READS
     1AS FOLLOWS--',5X,A2,A4,3X,6A4,23X,I4)
 4000 FORMAT(1H0,'TOO MANY ROOM CARDS INCLUDED. ALL ABOVE THE FIRST TEN
     1HAVE BEEN DELETED.',/' ','COURSE-',I2,5X,A2,A4,6A4,23X,I4)
 5000 FORMAT(1H0,'TOO MANY TEACHER CARDS INCLUDED. ALL ABOVE THE FIRST T
     1EN HAVE BEEN DELETED.',/1H ,'COURSE-',I2,5X,A2,A4,3X,6A4,23X,I4)
```

```
6000 FORMAT('OREQUEST WITH NO PERIODS CARD DETECTED--PDS SET TO ONE')
7000 FORMAT('OA COURSE THAT DOES NOT EXIST ON THE MASTER FILE WAS FOUND
    $, THE COURSE WAS IGNORED, COURSE NUMBER='I5)
8000 FORMAT('OTYPE CONFLICT IN COURSE',I5,/,' TYPE ACCEPTED IS ',
    1 A2,A4,3X,6A4,23X,I4)
C
C
C     THIS IS THE FIRST SECTION OF THE PROGRAM. IT READS THE CARDS AND
C     PREPARES THE INFORMATION ON THEM FOR THE NEXT SECTION.
C
      DO 10 I=1,2500
   10 TEMFIL(1,I) = 0
      READ(7'AV7) M
   25 READ(8,1000) CARD
      IF(CARD(2).EQ.COURSE) GO TO 50
      WRITE(6,2000) CARD
      GO TO 25
   50 CRSE=CARD(9)
      GO TO 800
   75 READ(8,1000,END=900) CARD
      IF(CARD(2).EQ.STUDEN) GO TO 100
      IF(CARD(2).EQ.PERIOD) GO TO 125
      IF(CARD(2).EQ.COURSE) GO TO 700
      IF(CARD(2).EQ.TEACH)  GO TO 400
76    IF(CARD(2).EQ.GROUP)  GO TO 350
      IF(CARD(2).EQ.ROOM)   GO TO 200
      IF(CARD(2).EQ.MAYSCH) GO TO 375
      IF(CARD(2).EQ.OPEN)   GO TO 300
      IF(CARD(2).EQ.OPMAY)  GO TO 325
      IF(CARD(2).EQ.BETPER) GO TO 225
      IF(CARD(2).EQ.ANDPER) GO TO 250
      IF(CARD(2).EQ.START)  GO TO 275
      IF (CARD(2).EQ.CONFI) GO TO 150
      IF(CARD(2).EQ.PRIOR)  GO TO 175
      IF(CARD(2).EQ.NOPRI)  GO TO 425
      IF(CARD(2).EQ.NOSHCR) GO TO 450
      IF(CARD(2).EQ.NOISHO) GO TO 475
      WRITE(6,3000) CRSE,CARD
      GO TO 75
  100 ZZZ= CARD(9)
      IF(ZZZ.EQ.0) GO TO 120
      IF(ZZZ.EQ.999) GO TO 115
      STUREQ=.TRUE.
      NUM=NUM+1
      IF(NUM.GE.2500) GO TO 75
      TEMPST(NUM)=ZZZ
      GO TO 75
  115 STUFIL=.TRUE.
      GO TO 75
  120 NOSTUD=.TRUE.
      GO TO 75
  125 PDS=CARD(9)
      GO TO 75
  150 CONMAX=CARD(9)
```

```
      GO TO 75
  175 PRI=PRI+1
      GO TO 75
  200 NN=NN+1
      ROMS=.TRUE.
      IF(NN.LE.10)GO TO 210
      WRITE(6,4000) CRSE,CARD
      NN=10
      GO TO 75
  210 ROOMS(NN)=CARD(9)
      GO TO 75
  225 BETPDS=CARD(9)
      GO TO 75
  250 ANDPDS=CARD(9)
      GO TO 75
  275 STARTS=CARD(9)
      GO TO 75
  300 OPENS=CARD(9)
      IF(GRP.OR.OP.OR.GRPMAY.OR.OPEMAY) GO TO 500
      OP=.TRUE.
      GG= OPENS
      TYPE = 3
      GO TO 75
  325 OPMAYS=CARD(9)
      IF(GRP.OR.OP.OR.GRPMAY.OR.OPEMAY) GO TO 500
      OPEMAY=.TRUE.
      GG= OPMAYS
      TYPE=5
      GO TO 75
  350 GROUPS=CARD(9)
      IF(GRP.OR.OP.OR.GRPMAY.OR.OPEMAY) GO TO 500
      GRP=.TRUE.
      GG= GROUPS
      TYPE=2
      GO TO 75
  375 GRMAYS=CARD(9)
      IF(GRP.OR.OP.OR.GRPMAY.OR.OPEMAY) GO TO 500
      GRPMAY=.TRUE.
      GG= GRMAYS
      TYPE = 4
      GO TO 75
  400 TT=TT+1
      IF(TT.LE.10) GO TO 410
      WRITE(6,5000) CRSE,CARD
      TT=10
      GO TO 75
  410 TECHER(TT)=CARD(9)
      GO TO 75
 425  IF(TYPE.EQ.0) TYPE=1
      IF(TYPE.GT.0) TYPE=-TYPE
      PRI=-1
      GO TO 75
  450 SHOCOD=SHOCOD-1
      GO TO 75
 475 SHOCOD=SHOCOD-2
     GO TO 75
500  WRITE(6,8000) CRSE,CARD
     OPENS=0
     OPMAYS=0
     GROUPS=0
     GRMAYS=0
     GRP=.FALSE.
     OP=.FALSE.
     GRPMAY=.FALSE.
     OPEMAY=.FALSE.
     GO TO 76
C
C    THIS IS THE SECOND SECTION OF THE PROGRAM. IT TAKES THE PREPARED
C    INFORMATION FROM SECTION ONE AND CREATES DATA SET 16 AND DATA SET
C    17.
C
  700 IF(M(CRSE).EQ.0) GO TO 950
      IF(STARTS.EQ.0) GO TO 701
      BETPDS= STARTS
      ANDPDS= STARTS
  701 CONTINUE
       IF (PDS.GT.0) GO TO 702
            PDS = 1
       WRITE(6,6000)
  702 CONTINUE
      AAA= 29-PDS
      IF(ANDPDS.EQ.0) ANDPDS=AAA
      IF(ANDPDS.GT.AAA) ANDPDS= AAA
      IF(BETPDS.GT.AAA) BETPDS=AAA
       IF(ANDPDS.GE.BETPDS) GO TO 703
       XTEMP = ANDPDS
           ANDPDS = BETPDS
       BETPDS = XTEMP
  703 CONTINUE
      IF(GRP.OR.GRPMAY.OR.OP.OR.OPEMAY) SHOCOD=0
      READ(10'M(CRSE))RMS,TCHS,STUNUM,(TEMSTD(MM),MM=1,STUNUM)
      IF(TECHER(1).NE.0) GO TO 705
      DO 704 MM=1,10
      IF(TCHS(MM).NE.0) TT=MM
  704 TECHER(MM)=TCHS(MM)
  705 PERIDS=PDS
      IF(OP.OR.OPEMAY) PERIDS=OPENS+OPMAYS
      IF(TYPE.EQ.1) PERIDS=0
      IF(GRP) PERIDS=GROUPS*PDS
      IF(GRPMAY) PERIDS=GRMAYS*PDS
      DO 710 MM=1,10
      TEMP=TECHER(MM)
      IF(TEMP.EQ.0) GO TO 711
  710 TEACHR(TEMP)=TEACHR(TEMP)+PERIDS
  711 IF (ROMS) GO TO 730
      DO 715 LL=1,10
      IF(RMS(LL).NE.0) NN=LL
  715 ROOMS(LL)=RMS(LL)
 730  TOTSTU = NUM
      KK = TOTSTU
      IF(STUREQ.AND..NOT.STUFIL) GO TO 741
      TOTSTU = NUM + STUNUM
      IF(TOTSTU.GT.2500) TOTSTU=2500
      KNUM=NUM+1
      KKK=0
      DO 740 KK=KNUM,TOTSTU
      KKK=KKK+1
 740  TEMPST(KK)=TEMSTD(KKK)
      IF(.NOT.STUREQ) GO TO 742
 741  CALL SORT(TEMPST,TOTSTU)
      CALL DUPCHK(TEMPST,TOTSTU)
      KK = TOTSTU
 742  IF(.NOT.(GRP.OR.OP.OR.GRPMAY.OR.OPEMAY)) GO TO 750
      DO 745 MM2=1,28
      KK=KK+1
 745  TEMPST(KK)=0
 750  NVORLT=OPENS+OPMAYS+GROUPS+GRMAYS
      NOM=NOM+1
      TEMFIL(1,NOM) = VA16
      TEMFIL(2,NOM) = CRSE
      IF(CONMAX.EQ.0) CONMAX=TOTSTU/3
      IF(TYPE.LE.1.AND.KK.EQ.0) TEMPST(1)=0
      WRITE(16'VA16) ROOMS,TECHER,PDS,CONMAX,SHOCOD,STARTS,BETPDS,ANDPDS
     1,NVORLT,TYPE,TOTSTU,CRSE,BLANK,(TEMPST(MM3),MM3=1,KK)
      TEMFIL(3,NOM) = PRIORY(PRI,TYPE,PDS,TOTSTU,STARTS,NN,TT,BETPDS,
     1 ANDPDS,GG)
 775  CRSE = CARD(9)
      IF(.NOT.FLAG) GO TO 800
 780  CALL DESCND(TEMFIL,NOM,DUMDUM)
      WRITE (17'VA17) (TEMFIL(1,K),K=1,2500)
      WRITE (17'VA17) (TEMFIL(2,K),K=1,2500)
      WRITE(17'VA17) TEACHR
      CALL EXIT
C     THIS IS THE THIRD SECTION OF THE PROGRAM. IT RESETS THE VARIABLES
C     USED IN SECTIONS ONE AND TWO TO THEIR ORIGINAL VALUES SO THAT
C     ANOTHER REQUEST CAN BE PROCESSED.
C
 800  NUM=0
      ROMS=.FALSE.
      STUFIL = .FALSE.
      NOSTUD=.FALSE.
      PDS=0
      CONMAX=0
      PRI=0
      NN=0
      DO 810 N=1,10
 810  ROOMS(N)=0
      BETPDS=1
      ANDPDS=0
      STARTS=0
      GG = 1
      OPENS=0
      OP=.FALSE.
      TYPE=0
      OPMAYS=0
      OPEMAY=.FALSE.
      GROUPS=0
      GRP=.FALSE.
      GRMAYS=0
      GRPMAY=.FALSE.
      TT=0
      DO 820 IM=1,10
  820 TECHER(IM) = 0
      SHOCOD=3
      PERIDS=0
      GO TO 75
  900 FLAG=.TRUE.
      GO TO 700
  950 WRITE(6,7000)CRSE
      CRSE= CARD(9)
      IF(FLAG) GO TO 780
      GO TO 800
C***********************************************************************
C***********************************************************************
      END
```

PRIORY

```
      FUNCTION PRIORY(PRI,TYPE,PDS,TOTSTU,STARTS,NN,BETPDS,ANDPDS,GG)
      IMPLICIT INTEGER*2 (A-Z)
      INTEGER*4 PRINUM,POW
      IF(PRI.GT.3) PRI=3
C     SET PRIORITY BLOCK
      EQUAL= PRI*8192
      PRINUM= EQUAL
C     SEPARATE BY TYPE
      IF(TYPE.EQ.3.OR.TYPE.EQ.5) GO TO 10
      PRINUM=PRINUM+ 2048
      IF(TYPE.NE.0.AND.TYPE.NE.1) GO TO 10
      PRINUM= PRINUM+ 2048
C     ADJUST BY NUMBER OF STUDENTS
   10 PRINUM=PRINUM+ TOTSTU/GG
C     ADJUST FOR NUMBER OF ROOMS
      PRINUM=PRINUM- 5*NN
C     ADJUST FOR NUMBER OF TEACHERS
      PRINUM=PRINUM+ 100*TT
C
      ADJUST FO START RESTRICTION
      IF(STARTS.NE.0) PRINUM=PRINUM+1024
C     IF PERIODS GREATER THAN 5 ADJUST FOR PERIODS
      POW= PDS*5
      IF(PDS.LT.6) POW= 0
      PRINUM= PRINUM+ POW
C
C     IF  SCHEDULEING RESTRICTIONS TIGHTER THAN NORMAL
C     ADJUST FOR THESE RESTRICTIONS
      POW= ANDPDS- BETPDS+ PDS-28
      IF(POW.NE.0) PRINUM=PRINUM+ PDS*(-POW)
C     TRIM THE PRIORITY NUMBER TO FIT THE PRIORITY BLOCK
      GREAT=(PRI+1)* 8192 - 1
      IF(PRINUM.GT.GREAT)  PRINUM= GREAT
      IF(PRINUM.LT.EQUAL) PRINUM= EQUAL
      PRIORY= PRINUM
      RETURN
      END
```

Job Step 8

SCHEDULE MAIN

```
C     MAIN CONTROL PROGRAM FOR SCHEDULING
      IMPLICIT INTEGER * 2 (A-Z) ,INTEGER*4($)
C
C
      COMMON /BLOK05/ DUMMY1(2),STDINF(28,256),DUMMY2(2504)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK10/ MATRIX(28)
      COMMON /BLOK30/ TEACRQ(500)
      COMMON /BLOKGH/ NOPDSD
C
C
      INTEGER*4 ROUND,RUMARY,TCHARY,RQST,MATRIX
C
C
      DIMENSION TCHRS(10),ROOMS(10),EXTRA(20),STDLST(2500),RQDATA(81)
C
C
      EQUIVALENCE (HOLDER(27),NUMOP),(HOLDER(21),NUMPDS),
     1(HOLDER(24),STRTPD),(HOLDER(11),TCHRS),(HOLDER,ROOMS),
     2(HOLDER(32),STDLST(1))(RQDATA(1),HOLDER(1))
 1004 FORMAT('0 THE PREVIOUS RESCHEDULE PASS HAD A PERCENT CHANGE OF',
     1 I5)
 1005 FORMAT('0FILLSKD PICKED UP',I10,'STUDENT PERIODS')
 1006 FORMAT('0PROGRAM FAILURE ON REQUEST',I5)
 1000 FORMAT ('0NO PRIORITY REQUEST',I5,' DENIED--FULL SCHEDULE')
 1001 FORMAT('0SCHEDULE OF REQUEST',I5,' WAS NOT ATTEMPTED BECAUSE OF',
     1 'INVALID REQUEST TYPE')
 1002 FORMAT('0THE FIRST ROUND OF RESCHEDULING HAS BEGUN,THE NUMBER',
     1 ' OF STUDENT PERIODS MISSED IS',I10)
 1003 FORMAT('0LAST RESCHEDULE IS FINISHED, FINAL PERCENT CHANGE WAS'
     1 ,I5)
C
C
      DEFINE FILE 16(600,162,L,AV16)
      DEFINE FILE 17(8,625,U,AV17)
      DEFINE FILE 25(40,896,U,AV25)
C
C
C     SET UP FILES TO BE READ ONLY ONCE
C
C
      REWIND 20
      READ (17'5) TEACRQ
      REQNUM = 0
      READ (17'1) RQST
      READ(17'7) DUMMY2
      OLDBPD=0
      BUMPD=0
      ROUND=1
      TEN=10
      TWO=2
      HOLDRQ=0
C
C     READ IN REQUEST INFORMATION
C
   10 REQNUM = REQNUM + 1
      K=RQST(REQNUM)
      IF (K.EQ.0) GO TO 900
      READ (16'K) RQDATA
      AV16=K+1
C
C     SEE IF MULTIPLE RECORDS NEEDED--READ IF NECESSARY
C
      IF (NUMSTU.GT.50) READ (16'AV16) (STDLST(I), I=51, NUMSTU)
C
C     CHECK  REQUEST TYPE AND CALL PROPER ROUTINE TO PROCESS
C
      IF(RQTYPE.LT.0) GO TO 800
  40  GO TO (100,90,400,50,390),ROUND
 50   IF(RQTYPE.EQ.3.OR.RQTYPE.EQ.5) PD=NUMOP
      IF(RQTYPE.EQ.2.OR.RQTYPE.EQ.4) PD=NUMOP*NUMPDS
      IF(RQTYPE.EQ.1.OR.RQTYPE.EQ.0) PD=NUMPDS
      I=TEACRQ(TCHRS(1))-PD
      IF(I.GE.0) GO TO 100
      WRITE(6,1000) REQNUM
      GO TO 10
 90   IF(RQTYPE.NE.0.OR.RQTYPE.NE.1) GO TO 10
      IF(STRTPD.GE.0) GO TO 10
      IF(ACSTRT.EQ.0) GO TO 91
      CALL DRPSKD
C     REARRANGE ROOMS BY USAGE
 91   DO 93 I=1,10
      J=I*2
      EXTRA(J)=ROOMS(I)
      IF(ROOMS(I).EQ.0) GO TO 92
      EXTRA(J-1) =SUM(RUMARY(ROOMS(I)))
      GO TO 93
 92   EXTRA(J-1) =30
 93   CONTINUE
      CALL BLKSRT(EXTRA,TEN,TWO)
      DO 95 I=1,10
      J=I*2
      ROOMS(I)=EXTRA(J)
 95   CONTINUE
 100  CONTINUE
      IF(RQTYPE.GE.0) GO TO 101
      WRITE(6,1001) REQNUM
      GO TO 10
 101  $RQTYP=RQTYPE
      IF(RQTYPE.EQ.0) $RQTYP=1
      GO TO (102,103,104,105,106),$RQTYP
 102  CALL RGSKED(+200)
```

```
      GO TO 200
 103  CALL GROUP
      GO TO 200
 104  CALL OPEN
      GO TO 200
 105  CALL MAY
      GO TO 200
 106  CALL OPENMY
 200  IF(RQTYPE.GT.1) GO TO 250
      IF(ACSTRT.LE.0) GO TO 220
      BUMPD=BUMPD+((NUMPDS-NUMSKD)*(NUMSTU))+(STRTPD*NUMSKD)
      GO TO 250
 220  BUMPD=BUMPD+(NUMSTU*NUMPDS)
C
C     WRITE MODIFIED RQDATA-- GO TO GET NEXT SET OF DATA
C
 250  WRITE(16'RQST(REQNUM)) RQDATA
      AV16=RQST(REQNUM)+1
      K = NUMSTU
      IF (RQTYPE.GT.1) K = NUMSTU + 28
      IF (K.GT.50) WRITE (16'AV16) (STDLST(I), I=51, K)
      IF(ACSTRT.LE.0) GO TO 10
      DO 300 I= 11,20
      A= RQDATA(I)
      IF(A.EQ.0) GO TO 10
 300  TEACRQ(A)= TEACRQ(A)- NUMSKD
      GO TO 10
 390  CONTINUE
 400  CONTINUE
      IF(RQTYPE.NE.0.OR.RQTYPE.NE.1) GO TO 10
      IF(STRTPD.GE.0) GO TO 10
      IF(ACSTRT.LE.0) GO TO 10
      CALL FILSKD
      WRITE(16'RQST(REQNUM)) RQDATA
      AV16=RQST(REQNUM)+1
      IF(NUMSTU.GT.50) WRITE(16'AV16)(STDLST(I),I=51,NUMSTU)
 800  GO TO (810,820,830,840,850),ROUND
 810  OLDBPD=BUMPD
      BUMPD=0
      ROUND=2
      REQNUM=0
      WRITE(6,1002) OLDBPD
      CALL RESET
      GO TO 10
 820  DIFF=OLDBPD-BUMPD
      PERDIF=(DIFF*100)/OLDBPD
      IF(PERDIF.GT.5) GO TO 825
      WRITE(6,1003) PERDIF
      NOPDS=0
      ROUND=3
      REQNUM=0
      GO TO 10
 825  OLDBPD=BUMPD
```

```
      BUMPD=0
      REQNUM=0
      WRITE(6,1004) PERDIF
       GO TO 10
830   WRITE(6,1005) NOPDSD
      ROUND=4
      REQNUM=REQNUM-1
       GO TO 10
840   RQTYPE=-RQTYPE
      IF(HOLDRQ.EQ.0) HOLDRQ=REQNUM
      GO TO 40
850   WRITE(6,1006) REQNUM
      GO TO 10
900   GO TO (810,820,920,910,920),ROUND
910   IF(HOLDRQ.EQ.0) GO TO 920
      REQNUM=HOLDRQ-1
      ROUND=5
       GO TO 10
920   CALL ENDIT
      END
```

GROUP

```
      SUBROUTINE GROUP
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK99/ SELECT(28), GRPCNT, COUNT, CODE, RFLAG
      LOGICAL * 1 RFLAG
      LOGICAL*1  TCHOK,SKEDOK
      DIMENSION  ROOMS(10), TCHRS(10), SLIST(2528)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),(HO
     1LDER(21),LONG), (HOLDER(25),BEGIN), (HOLDER(26),LAST), (HOLDER(27)
     2,TGRP), (HOLDER(29),TSTUD), (HOLDER(30), RLONG), (HOLDER(31), ACST
     3RT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
C
C
 1000 FORMAT(' GROUP REQUEST NOT SCHEDULED BECAUSE STUDENT COUNT = 0,
     1REQUEST NUMBER 'I7)
 1002 FORMAT(' GROUP REQUEST NOT SCHEDULED BECAUSE NO PERIODS WERE OPEN,
     * REQUEST NUMBER 'I7)
 1004 FORMAT(' GROUP REQUEST NOT SCHEDULED BECAUSE OF PROGRAM ERROR,
     1REQUEST NUMBER 'I7)
 1006 FORMAT(' GROUP REQUEST SCHEDULED, REQUEST NUMBER 'I7)
C
C
      IF(TSTUD.LE.0) GO TO 450
      CALL GROUPA
      GO TO (300,450,470,480),CODE
  300 CALL GROUPB
      GO TO 500
  450 WRITE(6,1000) REQNUM
      GO TO 600
  470 WRITE(6,1002) REQNUM
      GO TO 600
  480 WRITE(6,1004) REQNUM
      GO TO 600
  500 ACSTRT= 1
      WRITE(6,1006) REQNUM
      RETURN
  600 ACSTRT= 0
      RETURN
C*************************************************************************
C*************************************************************************
      END
```

m

ENDIT

```
      SUBROUTINE ENDIT
      IMPLICIT INTEGER*2 (A-Z)
      LOGICAL * 1 IN,UP
      INTEGER * 4 PART, TCHARY, RUMARY, RQST
      COMMON /BLOK05/ PART, STDINF(28,256), STDLST(2500), INSEC, IN, UP
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      END FILE 20
      REWIND 20
      IF(UP) WRITE(25'INSEC*4-3) STDINF
      REWIND 24
      WRITE(24) RUMARY
      END FILE 24
      REWIND 24
      WRITE(6,900)
  900 FORMAT ('0ALL DONE ')
      CALL EXIT
      END
```

GROUPA

```
      SUBROUTINE GROUPA
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK99/ SELECT(28), GRPCNT, COUNT, CODE, RFLAG
      LOGICAL*4 STDOK,STDOK2
      LOGICAL*1  TCHOK,SKEDOK
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      LOGICAL*1 SCHOOL(28), ROOMOK(28,10), RFLAG, FLAGZ, CHECK(100),
     1WCHECK(100)
      DIMENSION  ROOMS(10), TCHRS(10), SLIST(2528),HOUR(56)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),(HO
     1LDER(21),LONG), (HOLDER(25),BEGIN), (HOLDER(26),LAST), (HOLDER(27)
     2,TGRP), (HOLDER(29),TSTUD), (HOLDER(30), RLONG), (HOLDER(31), ACST
     3RT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C     SET UP VARIABLES FOR LATER USE IN PROGRAM.
      RLONG= LONG
      RFLAG= .FALSE.
      FLAGZ= .FALSE.
      KEEPB= BEGIN
C
      LSTART= LAST
      LAST= LAST+ LONG- 1
      RMCNT=0
      A= (LAST- BEGIN+ 1)/ LONG
      IF (TGRP.GT.A) TGRP= A
      DO 100 I=1,28
  100 SLIST(TSTUD+I)= -1
C
C     FIND WHICH PERIODS THE TEACHERS ARE FREE.
      DO 105 I=BEGIN,LSTART
  105 SCHOOL(I)= TCHOK(I,LONG)
C
C     FIND WHICH PERIODS ARE AVAILABLE IN EACH ROOM. FIND WHICH PERIODS
C     HAVE TEACHERS AND A ROOM AVAILABLE.
      DO 110 I=1,10
      IF (ROOMS(I).EQ.0) GO TO 113
      RMCNT= RMCNT+1
      DO 110 J= BEGIN,LSTART
  110 ROOMOK(J,I)= SKEDOK(RUMARY,ROOMS(I),J,LONG)
  113 IF(RMCNT.EQ.0)GO TO 121
      DO 120 I= BEGIN,LSTART
      IF (.NOT.SCHOOL(I)) GO TO 120
      DO 115 J=1,RMCNT
      IF (ROOMOK(I,J)) GO TO 120
  115 CONTINUE
      SCHOOL(I)= .FALSE.
  120 CONTINUE
      GO TO 123
  121 RFLAG= .TRUE.
  123 RECORD= 0
C
C     SET UP STUDENT COUNTER.
      COUNT= TSTUD
      IF (COUNT.LE.0) GO TO 450
      IF (COUNT.GT.100) COUNT= 100
      DO 125 I=1,LSTART
      A= I*2
      HOUR(A)= 0
  125 HOUR(A-1)= I
C
C     SET UP AN ARRAY OF 100 STUDENTS AND THE PERIODS THAT EACH IS
C     AVAILABLE AND FIND A TOTAL FOR EACH PERIOD.
      DO 135 I= 1,COUNT
      DAYZ(I)= 0
      IF (.NOT.SCHOOL(BEGIN)) GO TO 130
      IF (.NOT.STDOK(SLIST(I),BEGIN,LONG)) GO TO 130
      CALL PLACE(DAYZ,I,BEGIN,ONE)
      B= BEGIN* 2
      HOUR(B)= HOUR(B) +1
  130 IF (BEGIN.EQ.LSTART) GO TO 136
      A= BEGIN+ 1
      DO 135 J=A,LSTART
      IF (.NOT.SCHOOL(J)) GO TO 135
      IF (.NOT.STDOK(SLIST(I),J,LONG)) GO TO 135
      CALL PLACE (DAYZ,I,J,ONE)
C
C     SET UP VARIABLES FOR LATER USE IN THE PROGRAM
      B= J*2
      HOUR(B)= HOUR(B)+1
  135 CONTINUE
  136  BEGIN=29
      LAST= 0
      GRPCNT= 0
      DO 137 I=1,COUNT
  137 CHECK(I)= .FALSE.
C
C     SORT PERIODS IN DESCENDING ORDER BY NUMBER OF STUDENTS AVAILABLE
C     IN EACH PERIOD.
      CALL DESCND (HOUR,LSTART,TWO)
C
C     COUNT NUMBER OF STARTING TIMES AVAILABLE. IF NONE AVAILABLE TRANS
C     FER.
      STCNT= 0
      DO 140 I= 1,LSTART
  140 IF (HOUR(I*2).GT.0) STCNT= STCNT+1
      IF (STCNT.LE.0) GO TO 470
C
```

```
C     SCHEDULE FIRST PERIOD AVAILABLE.
      PERIOD= HOUR(1)
C
C     SCHEDULE TEACHERS, ROOM AND REQUEST NUMBER.
  143 DO 145 I= 1,10
      A= TCHRS(I)
      IF (A.EQ.0) GO TO 146
  145 CALL PLACE (TCHARY,   A    ,PERIOD,LONG)
  146 CALL PLACE(ROST,RONUM,PERIOD,LONG)
      IF (PERIOD.LT.BEGIN) BEGIN= PERIOD
      IF (PERIOD.GT.LAST) LAST= PERIOD
      IF (RFLAG) GO TO 153
C
C     FIND WHICH ROOM AVAILABLE.
      DO 150 I= 1,RMCNT
      IF (ROOMOK(PERIOD,I)) GO TO 151
  150 CONTINUE
      GO TO 480
 151  CALL PLACE(RUMARY,ROOMS(I),PERIOD,LONG)
      A= TSTUD+ PERIOD- 1
      DO 152 J=1,LONG
  152 SLIST(A+J)= ROOMS(I)
C
C     CHECK IF THE TOTAL NUMBER OF PERIODS REQUESTED HAS BEEN REACHED.
  153 GRPCNT= GRPCNT+ 1
      SELECT(GRPCNT)= PERIOD
      IF (GRPCNT.GE.TGRP) GO TO 300
      IF(STCNT.EQ.1) GO TO 300
C
C     CHECK TEACHER SCHEDULE FOR CHANGES CAUSED BY SCHEDULING LAST
C     GROUP.
      DO 155 I= KEEPB,LSTART
  155 SCHOOL(I)= SCHOOL(I).AND.TCHOK(I,LONG)
C
C     SET UP VARIABLES THAT COUNT HOW MANY STUDENTS ARE AVAILABLE FOR
C     AT LEAST ONE PERIOD.
      DO 162 I=1,COUNT
  162 CHECK(I)= CHECK(I).OR..NOT.SKEDOK(DAYZ,I,PERIOD,ONE)
      IF  (FLAGZ) GO TO 180
C
C
C     FIND THE NEXT BEST GROUP TO FIT ALREADY SELECTED GROUPS.
C
      KEEPZ= 101
      DO 170 I=2,STCNT
      A= HOUR(I*2-1)
      IF (.NOT.SCHOOL(A)) GO TO 170
```

```
C
C     COUNT NUMBER OF STUDENTS WHO WOULD NOT BE AVAILABLE TO BE
C     SCHEDULED WITH THIS GROUP AND ALREADY SCHEDULED GROUPS.
      ZERO= 0
      DO 165 J= 1,COUNT
      WCHECK(J)= CHECK(J).OR..NOT.SKEDOK(DAYZ,J,A,ONE)
  165 IF (.NOT.WCHECK(J)) ZERO= ZERO+ 1
C
C     IF ALL STUDENTS ACCOUNTED FOR TRANSFER.
      IF (ZERO.LE.0) GO TO 175
C
C     CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
      IF (ZERO.GE.KEEPZ) GO TO 170
      KEEPZ= ZERO
      KEEPP= A
  170 CONTINUE
C
C     TRANSFER TO SCHEDULING IF NO SCHEDULABLE GROUPS ARE FOUND.
      IF (KEEPZ.EQ.101) GO TO 300
      PERIOD= KEEPP
      GO TO 143
C
C     TRANSFER HERE IF ALL STUDENTS ARE ACCOUNTED FOR IN AT LEAST ONE
  175 FLAGZ= .TRUE.
C     PERIOD.
      PERIOD= A
      GO TO 143
  180 KEEPT= 0
C
C     FIND STARTING TIME WITH THE HIGHEST TOTAL STUDENTS AVAILABLE THAT
C     WILL FIT GROUPS ALREADY SCHEDULED.
      DO 185 I=2,STCNT
      A= I*2
      IF(.NOT.SCHOOL(HOUR(A-1)).OR.HOUR(A).LE.KEEPT) GO TO 185
      KEEPT= HOUR(A)
      KEEPP= HOUR(A-1)
  185 CONTINUE
      IF (KEEPT.EQ.0) GO TO 300
C
C     SCHEDULE TEACHERS, ROOM AND REQUEST NUMBER.
      DO 187 I= 1, 10
      A= TCHRS(I)
      IF (A.EQ.0) GO TO 188
  187 CALL PLACE (TCHARY,  A     ,KEEPP,LONG)
  188 CALL PLACE (RQST,RQNUM,KEEPP,LONG)
      IF(RFLAG) GO TO 192
      DO 189 I= 1,RMCNT
      IF(ROOMOK(KEEPP,I)) GO TO 190
  189 CONTINUE
      GO TO 480
 190  CALL PLACE(RUMARY,ROOMS(I),KEEPP ,LONG)
C
C     SET UP VARIABLES FOR OUTPUT ROUTINES.
      A= TSTUD+ KEEPP- 1
      DO 191 J= 1,LONG
  191 SLIST(A+J)= ROOMS(I)
  192 GRPCNT= GRPCNT+ 1
      SELECT(GRPCNT)= KEEPP
      IF(KEEPP.LT.BEGIN) BEGIN= KEEPP
      IF(KEEPP.GT.LAST) LAST= KEEPP
      IF (GRPCNT.GE.TGRP) GO TO 300
C
C     CHECK TEACHER SCHEDULE FOR CHANGES CAUSED BY SCHEDULING LAST
C     GROUP.
      DO 193 I= KEEPB, LSTART
  193 SCHOOL(I)= SCHOOL(I).AND.TCHOK(I,LONG)
      GO TO 180
  300 CODE= 1
      RETURN
  450 CODE= 2
      RETURN
  470 CODE= 3
      RETURN
  480 CODE= 4
      RETURN
C*********************************************************************
C*********************************************************************
      END
```

GROUPB

```
      SUBROUTINE GROUPB
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),ROST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK99/ SELECT(28), GRPCNT, COUNT, CODE, RFLAG
      LOGICAL* 1 RFLAG
      LOGICAL*4 STDOK
      LOGICAL*1  TCHOK,SKEDOK
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,ROST
      DIMENSION HOUR(29),ROOMS(10),TCHRS(10),SLIST(2528), STDENT(2,10
     10), STUDIN(28)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),(HO
     1LDER(21),LONG), (HOLDER(25),BEGIN), (HOLDER(26),LAST), (HOLDER(27)
     2,TGRP), (HOLDER(29),TSTUD), (HOLDER(30), RLONG), (HOLDER(31), ACST
     3RT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
      NAMELIST/ANAME/ A,B,COUNT,KEEPH,KEEPN,KEEPT,I,RECORD,GRPCNT,
     1 STUDIN,SELECT,HOUR,STDENT
C
C
C
C        **SCHEDULE STUDENTS HERE**
C
  300 IF (GRPCNT.LE.1) GO TO 430
C
C     SORT STARTING TIMES IN CONSECUTIVE ORDER BY PERIOD.
      CALL BLKSRT(SELECT,GRPCNT,ONE)
C
C     SET UP VARIABLES FOR LATER USE IN PROGRAM.
      RECORD= 0
      DO 301 I=1,28
  301 STUDIN(I)= 0
  302 DO 303 I=1,29
  303  HOUR(I)= 0
C
C     SET UP AN ARRAY OF 100 STUDENTS AND THE PERIODS THAT EACH IS
C     AVAILABLE AND FIND A TOTAL FOR EACH PERIOD.
      DO 315 I= 1,COUNT
      A= RECORD* 100+ I
      DAYZ(I)= 0
      STDENT(1,I)= I
      STDENT(2,I)= 0
      IF (.NOT.STDOK(SLIST(A),SELECT(1),LONG))GO TO 310
      CALL PLACE(DAYZ,I,ONE,ONE)
      HOUR(1)= HOUR(1)+ 1
      STDENT(2,I)= STDENT(2,I)+ 1
  310 DO 315 J=2,GRPCNT
```

```
      IF(.NOT.STDOK(SLIST(A),SELECT(J),LONG)) GO TO 315
      CALL PLACE(DAYZ,I,J,ONE)
      HOUR(J)= HOUR(J)+1
      STDENT(2,I)= STDENT(2,I)+1
  315 CONTINUE
C
C     SORT STUDENTS BY NUMBER OF PERIODS FOR WHICH EACH STUDENT IS
C     AVAILABLE.
      CALL BLKSRT(STDENT,COUNT,TWO)
C
C     FIND THE STARTING TIME WITH THE LOWEST NUMBER OF STUDENTS ABLE TO
C     BE SCHEDULED AND IF TIE, FIND THE LOWEST NUMBER OF STUDENTS
C     ALREADY SCHEDULED FOR THAT PERIOD.
      DO 360 I= 1,COUNT
      A= STDENT(1,I)
      B= RECORD* 100+ A
      IF (STDENT(2,I).EQ.0) GO TO 350
      KEEPH = 201
      KEEPN= 3000
      KEEPT= 29
      DO 330 J= 1,GRPCNT
      IF (SKEDOK(DAYZ,A,J          ,ONE)) GO TO 330
C
C     COMPARE AGAINST BEST PERIOD SO FAR.
      IF ((HOUR(J).LT.KEEPH).OR.(HOUR(J).EQ.KEEPH.AND.STUDIN(J).LT.KEEPN
     1)) GO TO 325
C
C     SUBTRACT ONE FROM THE NUMBER OF STUDENTS STILL ABLE TO BE
C     SCHEDULED AT THIS PERIOD IF PERIOD NOT SCHEDULED.
      HOUR(J)= HOUR(J)-1
      GO TO 330
C
C     SCHEDULED AT THIS PERIOD.
C     SUBTRACT ONE FROM THE NUMBER OF STUDENTS STILL ABLE TO BE
  325 HOUR(KEEPT)= HOUR(KEEPT)- 1
      KEEPH= HOUR(J)
      KEEPN= STUDIN(J)
      KEEPT= J
  330 CONTINUE
      IF(KEEPT.GT.28) GO TO 340
C
C     SCHEDULE STUDENT AND ADJUST THE PARAMETERS.
      CALL STCHNG(SLIST(B), SELECT(KEEPT), LONG, REQNUM)
      HOUR(KEEPT)= HOUR(KEEPT)+ 1
      STUDIN(KEEPT)=STUDIN(KEEPT)+ 1
      GO TO 360
  340 WRITE(6,ANAME)
C
```

```
C       IF STUDENT NOT SCHEDULED, FLAG SET FOR OUTPUT ROUTINES
  350 SLIST(B)= SLIST(B)+ 24576
  360 CONTINUE
C       CHECK IF MORE STUDENTS STILL TO BE SCHEDULED.
      RECORD= RECORD+ 1
      A= TSTUD- RECORD* 100
      IF (A.LE.0) GO TO 400
      COUNT= A
      IF (COUNT.GT.100) COUNT= 100
      GO TO 302
C
C       ADJUST VARIABLES FOR OUTPUT IF NO ROOMS ARE NEEDED.
  400 IF (.NOT.RFLAG) GO TO 410
      DO 405 I= 1,28
  405 SLIST(TSTUD+I)=0
C
C       CHECK TO SEE IF PERIOD WAS USED. IF NOT, DELETE TEACHERS, ROOM
C       AND REQUEST NUMBER.
  410 DO 415 I= 1,GRPCNT
      IF (STUDIN(I).GT.0) GO TO 415
      A= SELECT(I)
      DO 412 J=1, 10
      C= TCHRS(J)
      IF (C.EQ.0) GO TO 413
  412 CALL DELETE(TCHARY,  C       ,A,LONG)
  413 CALL DELETE(RQST,RQNUM,A,LONG)
      IF (RFLAG) GO TO 415
      B= TSTUD+ A
      CALL DELETE(RUMARY,SLIST(B),A,LONG)
      SLIST(B)=-1
  415 CONTINUE
      GO TO 500
C
C
C       IF NUMBER OF GROUPS WAS ONE, TRANSFER HERE.
C
  430 A= SELECT(1)
      STUDIN(1)=0
C
C       CHECK IF STUDENT IS AVAILABLE. IF SO, SCHEDULE.
      DO 435 I=1,TSTUD
      IF (.NOT.STDOK(SLIST(I),A,LONG)) GO TO 435
C
C       ADJUST VARIABLES FOR OUTPUT ROUTINES.
      STUDIN(1)= STUDIN(1)+ 1
      CALL STCHNG(SLIST(I),A,LONG,REQNUM)
  435 CONTINUE
      BEGIN=A
      LAST= A
C
```

```
C       CHECK IF PERIOD WAS USED.
        IF(STUDIN(1).LE.0)GO TO 400
C
C       ADJUST VARIABLES FOR OUTPUT.
        IF (.NOT.RFLAG) GO TO 500
        DO 440 I= 1,28
  440   SLIST(TSTUD+I)=0
        GO TO 500
  500   RETURN
C*****************************************************************************
C*****************************************************************************
        END
```

RESET

```
      SUBROUTINE RESET
C     THIS IS A ROUTINE TO SET THE ARRAY MATRIX TO ONES
      IMPLICIT INTEGER * 2 (A-Z)
C     THIS SUBROUTINE TURNS ON BITS IN THE ARRAY MATRIX
C     THIS IS DONE SO THAT RGSKED WILL MAKE A MORE THUROUGH SEARCH.
      COMMON/BLOK10/ MATRIX(28)
      INTEGER*4 MATRIX
      DATA NUM/Z0FFFFFFF/
      DO 10 I=1,28
      MATRIX(I)=NUM
 10   CONTINUE
      RETURN
      END
```

MAY

```
      SUBROUTINE MAY
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      LOGICAL*4 STDOK
      LOGICAL*1  TCHOK,SKEDOK
      LOGICAL*1 SCHOOL(28), ROOMOK(28,10), RFLAG, FLAGZ, CHECK(100),
     1WCHECK(100), SCAT
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION HOUR(56), ROOMS(10), TCHRS(10), $LIST(2528), STDENT(2,10
     10), SELECT(28), STUDIN(28), OUT(10)
      EQUIVALENCE(KEEPT,KEEPZ),(HOLDER(1),ROOMS(1)),(HOLDER(11),
     1 TCHRS(1)),(HO
     1LDER(21),LONG), (HOLDER(25),BEGIN), (HOLDER(26),LAST), (HOLDER(27)
     2,TGRP), (HOLDER(29),TSTUD), (HOLDER(30), RLONG), (HOLDER(31), ACST
     3RT), (HOLDER(32),SLIST(1)),(OUT(1)STNUM),(OUT(2),STANUM),
     4(OUT(4),RNUM),(OUT(3),CNUM)
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
      DATA OUT/10*0/
C
C
 1000 FORMAT(' MAY REQUEST NOT SCHEDULED BECAUSE STUDENT COUNT = 0,
     *REQUEST NUMBER ',I7)
 1002 FORMAT(' MAY REQUEST NOT SCHEDULED BECAUSE NO PERIODS WERE OPEN,
     1REQUEST NUMBER ',I7)
 1004 FORMAT(' MAY REQUEST NOT SCHEDULED BECAUSE OF PROGRAM ERROR,
     *REQUEST NUMBER ',I7)
 1006 FORMAT(' MAY REQUEST SCHEDULED, REQUEST NUMBER ',I7)
 1008 FORMAT(' MAY REQUEST NOT SCHEDULED BECAUSE NO ROOMS WERE'
     *' ASSIGNED TO REQUEST, REQUEST NUMBER ',I7)
C
C
      IF(TSTUD.LE.0) GO TO 450
C
C     SET UP VARIABLES FOR LATER USE IN PROGRAM.
      CNUM= RLONG
      RLONG= LONG
      RFLAG= .FALSE.
      FLAGZ= .FALSE.
      KEEPB= BEGIN
C
      LSTART= LAST
      LAST= LAST+ LONG- 1
      RMCNT=0
      A= (LAST- BEGIN+ 1)/ LONG
      IF (TGRP.GT.A) TGRP= A
      DO 100 I=1,28
```

```
  100 SLIST(TSTUD+I)= -1
C
C     FIND WHICH PERIODS THE TEACHERS ARE FREE.
      DO 105 I=BEGIN,LSTART
  105 SCHOOL(I)= TCHOK(I,LONG)
C
C     FIND WHICH PERIODS ARE AVAILABLE IN EACH ROOM. FIND WHICH PERIODS
C     HAVE TEACHERS AND A ROOM AVAILABLE.
      DO 110 I=1,10
      IF (ROOMS(I).EQ.0) GO TO 113
      RMCNT= RMCNT+1
      DO 110 J= BEGIN,LSTART
  110 ROOMOK(J,I)= SKEDOK(RUMARY,ROOMS(I),J,LONG)
  113 IF(RMCNT.EQ.0) GO TO 550
      DO 120 I= BEGIN,LSTART
      IF (.NOT.SCHOOL(I)) GO TO 120
      DO 115 J=1,RMCNT
      IF (ROOMOK(I,J)) GO TO 120
  115 CONTINUE
      SCHOOL(I)= .FALSE.
  120 CONTINUE
      GO TO 123
  121 RFLAG=.TRUE.
  123 RECORD= 0
C
C     SET UP STUDENT COUNTER.
      COUNT= TSTUD
      IF (COUNT.GT.100) COUNT= 100
      DO 125 I=1,LSTART
      A= I*2
      HOUR(A)= 0
  125 HOUR(A-1)= I
C
C     SET UP AN ARRAY OF 100 STUDENTS AND THE PERIODS THAT EACH IS
C     AVAILABLE AND FIND A TOTAL FOR EACH PERIOD.
      DO 135 I= 1,COUNT
      DAYZ(I)= 0
      IF (.NOT.SCHOOL(BEGIN)) GO TO 130
      IF (.NOT.STDOK(SLIST(I),BEGIN,LONG)) GO TO 130
      CALL PLACE(DAYZ,I,BEGIN,ONE)
      B= BEGIN* 2
      HOUR(B)= HOUR(B) +1
  130 IF (BEGIN.EQ.LSTART) GO TO 136
      A= BEGIN+ 1
      DO 135 J=A,LSTART
      IF (.NOT.SCHOOL(J)) GO TO 135
      IF (.NOT.STDOK(SLIST(I),J,LONG)) GO TO 135
      CALL PLACE (DAYZ,I,J,ONE)
C
C     SET UP VARIABLES FOR LATER USE IN THE PROGRAM
      B= J*2
      HOUR(B)= HOUR(B)+1
  135 CONTINUE
  136  BEGIN=29
```

```
      LAST= 0
      GRPCNT= 0
      DO 137 I=1,COUNT
  137 CHECK(I)= .FALSE.
C
C     SORT PERIODS IN DESCENDING ORDER BY NUMBER OF STUDENTS AVAILABLE
C     IN EACH PERIOD.
      CALL DESCND (HOUR,LSTART,TWO)
C
C     COUNT NUMBER OF STARTING TIMES AVAILABLE. IF NONE AVAILABLE TRANS
C     FER.
      STCNT= 0
      DO 140 I= 1,LSTART
  140 IF (HOUR(I*2).GT.0) STCNT= STCNT+1
      IF (STCNT.LE.0) GO TO 470
C
C     SCHEDULE FIRST PERIOD AVAILABLE.
      PERIOD= HOUR(1)
C
C     SCHEDULE TEACHERS, ROOM AND REQUEST NUMBER.
  143 DO 145 I= 1,10
      A= TCHRS(I)
      IF (A.EQ.0) GO TO 146
  145 CALL PLACE (TCHARY,    A     ,PERIOD,LONG)
  146 CALL PLACE(RQST,RQNUM,PERIOD,LONG)
      IF (PERIOD.LT.BEGIN) BEGIN= PERIOD
      IF (PERIOD.GT.LAST) LAST= PERIOD
      IF (RFLAG) GO TO 153
C
C     FIND WHICH ROOM AVAILABLE.
      DO 150 I= 1,RMCNT
      IF (ROOMOK(PERIOD,I)) GO TO 151
  150 CONTINUE
      GO TO 480
 151  CALL PLACE(RUMARY,ROOMS(I),PERIOD,LONG)
      A= TSTUD+ PERIOD- 1
      DO 152 J=1,LONG
  152 SLIST(A+J)= ROOMS(I)
C
C     CHECK IF THE TOTAL NUMBER OF PERIODS REQUESTED HAS BEEN REACHED.
  153 GRPCNT= GRPCNT+ 1
      SELECT(GRPCNT)= PERIOD
      IF (GRPCNT.GE.TGRP) GO TO 300
      IF(STCNT.EQ.1) GO TO 300
C
C     CHECK TEACHER SCHEDULE FOR CHANGES CAUSED BY SCHEDULING LAST
C     GROUP.
      DO 155 I= KEEPB,LSTART
  155 SCHOOL(I)= SCHOOL(I).AND.TCHOK(I,LONG)
C
C     SET UP VARIABLES THAT COUNT HOW MANY STUDENTS ARE AVAILABLE FOR
C     AT LEAST ONE PERIOD.
      DO 162 I=1,COUNT
  162 CHECK(I)= CHECK(I).OR..NOT.SKEDOK(DAYZ,I,PERIOD,ONE)
      IF  (FLAGZ) GO TO 180
```

```
C
C
C     FIND THE NEXT BEST GROUP TO FIT ALREADY SELECTED GROUPS.
C
      KEEPZ= 101
      DO 170 I=2,STCNT
      A= HOUR(I*2-1)
      IF (.NOT.SCHOOL(A)) GO TO 170
C
C     COUNT NUMBER OF STUDENTS WHO WOULD NOT BE AVAILABLE TO BE
C     SCHEDULED WITH THIS GROUP AND ALREADY SCHEDULED GROUPS.
      ZERO= 0
      DO 165 J= 1,COUNT
      WCHECK(J)= CHECK(J).OR..NOT.SKEDOK(DAYZ,J,A,ONE)
  165 IF (.NOT.WCHECK(J)) ZERO= ZERO+ 1
C
C     IF ALL STUDENTS ACCOUNTED FOR TRANSFER.
      IF (ZERO.LE.0) GO TO 175
C
C     CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
      IF (ZERO.GE.KEEPZ) GO TO 170
      KEEPZ= ZERO
      KEEPP= A
  170 CONTINUE
C
C     TRANSFER TO SCHEDULING IF NO SCHEDULABLE GROUPS ARE FOUND.
      IF (KEEPZ.EQ.101) GO TO 300
      PERIOD= KEEPP
      GO TO 143
  175 FLAGZ= .TRUE.
C
C     TRANSFER HERE IF ALL STUDENTS ARE ACCOUNTED FOR IN AT LEAST ONE
C     PERIOD.
      PERIOD= A
      GO TO 143
  180 KEEPT= 0
C
C     FIND STARTING TIME WITH THE HIGHEST TOTAL STUDENTS AVAILABLE THAT
C     WILL FIT GROUPS ALREADY SCHEDULED.
      DO 185 I=2,STCNT
      A= I*2
      IF(.NOT.SCHOOL(HOUR(A-1)).OR.HOUR(A).LE.KEEPT) GO TO 185
      KEEPT= HOUR(A)
      KEEPP= HOUR(A-1)
  185 CONTINUE
      IF (KEEPT.EQ.0) GO TO 300
C
C     SCHEDULE TEACHERS, ROOM AND REQUEST NUMBER.
      DO 187 I= 1, 10
      A= TCHRS(I)
      IF (A.EQ.0) GO TO 188
  187 CALL PLACE (TCHARY,  A      ,KEEPP,LONG)
  188 CALL PLACE (RQST,RQNUM,KEEPP,LONG)
      IF(RFLAG) GO TO 192
      DO 189 I= 1,RMCNT
      IF (ROOMOK(KEEPP,I)) GO TO 190
  189 CONTINUE
      GO TO 480
 190  CALL PLACE(RUMARY,ROOMS(I),KEEPP ,LONG)
C
C     SET UP VARIABLES FOR OUTPUT ROUTINES.
      A= TSTUD+ KEEPP- 1
      DO 191 J= 1,LONG
  191 SLIST(A+J)= ROOMS(I)
  192 GRPCNT= GRPCNT+ 1
      SELECT(GRPCNT)= KEEPP
      IF(KEEPP.LT.BEGIN) BEGIN= KEEPP
      IF(KEEPP.GT.LAST) LAST= KEEPP
      IF (GRPCNT.GE.TGRP) GO TO 300
C
C     CHECK TEACHER SCHEDULE FOR CHANGES CAUSED BY SCHEDULING LAST
C     GROUP.
      DO 193 I= KEEPB, LSTART
  193 SCHOOL(I)= SCHOOL(I).AND.TCHOK(I,LONG)
      GO TO 180
C
C
C
C        **SCHEDULE STUDENTS HERE**
C
  300 SCAT= .FALSE.
      IF (GRPCNT.EQ.1) SCAT=.TRUE.
      DO 301 I= 1,GRPCNT
  301 STUDIN(I)= 0
C
C     CHECK TO SEE IF STUDENTS ARE AVAILABLE THIS PERIOD. IF NOT
C     TRANSFER.
      DO 325 I= 1,TSTUD
      TIME= SELECT(1)
      STNUM= SLIST(I)
      IF (.NOT.STDOK(STNUM,TIME,LONG)) GO TO 310
      STUDIN(1)= STUDIN(1)+ 1
      ASSIGN 310 TO $WITCH
      GO TO 330
  310 IF (SCAT) GO TO 325
      DO 320 J= 2,GRPCNT
      TIME= SELECT(J)
      IF(.NOT.STDOK(STNUM,TIME,LONG)) GO TO 320
      STUDIN(J)= STUDIN(J)+ 1
      ASSIGN 320 TO $WITCH
      GO TO 330
  320 CONTINUE
  325 CONTINUE
      GO TO 410
  330 STANUM= TIME* 32+ LONG
C
C     PREPARE VARIABLES TO WRITE OUT MAY RECORD.
      RNUM= SLIST(TSTUD+ TIME )
      IF (RFLAG) RNUM=0
      WRITE (20) OUT
      GO TO SWITCH, (310,320)
C
C     ADJUST VARIABLES FOR OUTPUT IF NO ROOMS ARE NEEDED.
  400 IF (.NOT.RFLAG) GO TO 410
      DO 405 I= 1,28
  405 SLIST(TSTUD+I)=0
C
C     CHECK TO SEE IF PERIOD WAS USED. IF NOT, DELETE TEACHERS, ROOM
C     AND REQUEST NUMBER.
  410 DO 415 I= 1,GRPCNT
      IF (STUDIN(I).GT.0) GO TO 415
      A= SELECT(I)
      DO 412 J=1, 10
      C= TCHRS(J)
      IF (C.EQ.0) GO TO 413
  412 CALL DELETE(TCHARY,  C      ,A,LONG)
  413 CALL DELETE(RQST,RQNUM,A,LONG)
      IF (RFLAG) GO TO 415
      B= TSTUD+ A
      CALL DELETE(RUMARY,SLIST(B),A,LONG)
      SLIST(B)=-1
  415 CONTINUE
      GO TO 500
  450 WRITE(6,1000) REQNUM
      GO TO 600
  470 WRITE(6,1002) REQNUM
      GO TO 600
  480 WRITE(6,1004) REQNUM
      GO TO 600
  500 ACSTRT= 1
      WRITE(6,1006) REQNUM
      RETURN
  550 WRITE(6,1008) REQNUM
  600 ACSTRT= 0
      RETURN
C*********************************************************************
C*********************************************************************
      END
```

OPENA

```
      SUBROUTINE OPENA
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK77/DAYZ,STARTS,STDENT,HOUR,SELECT,RMCNT,COUNT,KEEPZ,
     1DIGIT,PNUM,RTAG,ROOMOK
      LOGICAL*1  TCHOK,SKEDOK,STDOK,STDOK2
      LOGICAL*1 STAFF(28), ROOMOK(28,10), CHECK(100), PERIOD(28), RTAG
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2528), HOUR(29), SELECT(28),
     1STARTS(840), STDENT(200)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
C
C     SET UP VARIABLES FOR USE LATER IN THIS PROGRAM.
      RLONG= LONG
      A= LAST+ LONG- BEGIN
      IF(TOPEN.GT.A) TOPEN=A
      RTAG= .TRUE.
      RMCNT= 0
      DO 103 I=1,28
  103 SLIST(TSTUD+I)= -1
C
C     FIND WHEN TEACHERS ARE AVAILABLE.
      DO 105 I=BEGIN,LAST
C
C     FIND WHEN ROOMS ARE AVAILABLE.
  105 STAFF(I)= TCHOK(I,LONG)
      DO 115 I=1,10
      IF (ROOMS(I).EQ.0) GO TO 117
      RMCNT=RMCNT+1
      DO 115 J=BEGIN,LAST
  115 ROOMOK(J,I)= SKEDOK(RUMARY,ROOMS(I),J,LONG).AND.STAFF(J)
  117 IF (RMCNT.NE.0) GO TO 120
C
C     IF NO ROOMS REQUESTED TRANSFER HERE.
      RMCNT=1
      DO 118 I=BEGIN,LAST
  118 ROOMOK(I,1)= STAFF(I)
      RTAG= .FALSE.
C
```

```
C      SET UP STUDENT COUNTER.
  120 COUNT= TSTUD
      IF (COUNT.LE.0) RETURN
      IF (COUNT.GT.100) COUNT=100
      DO 125 I= 1,29
  125 HOUR(I)= 0
C
C      USE FIRST 100 STUDENTS TO BUILD AN ARRAY OF STUDENTS BY PERIOD
C      AND COUNT HOW MANY STUDENTS ARE AVAILABLE EACH PERIOD.
      DO 130 I= 1,COUNT
      DAYZ(I)=0
      IF(.NOT.STDOK(SLIST(I),BEGIN,LONG)) GO TO 127
      CALL PLACE (DAYZ,I,BEGIN,ONE)
      HOUR(BEGIN)= HOUR(BEGIN)+1
  127 A= BEGIN+1
      IF (A.GT.LAST) GO TO 130
      DO 129 J=A,LAST
      IF(.NOT.STDOK(SLIST(I),J     ,LONG)) GO TO 129
      CALL PLACE (DAYZ,I,J,ONE)
      HOUR(J)= HOUR(J)+1
  129 CONTINUE
  130 CONTINUE
      RETURN
C***********************************************************************
C***********************************************************************
      END
```

FILSKD

```
      SUBROUTINE FILSKD
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
C
      COMMON /BLOK05/ PART, STDINF(28,256), DUMMY2(2500), INSEC, IN, UP
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK30/ TEACRQ(500)
      COMMON/BLOKGH/ NOPDSD
C
      DIMENSION STDLST(2500),FLAG(7)
C
C
      LOGICAL*1 IN,UP
C
      INTEGER*4 POSNUM,PART,MAGIC,CHOICE
      INTEGER*4  RUMARY, TCHARY,RQST
C
C
      EQUIVALENCE (HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1 (HOLDER(21), NUMPDS), (HOLDER(22), CONMAX), (HOLDER(23), SHRTCD),
     2 (HOLDER(24), START), (HOLDER(25), BETPDS), (HOLDER(26), ANDPDS),
     3 (HOLDER(29), NUMSTU), (HOLDER(30), NUMSKD), (HOLDER(31), ACSTRT),
     4 (HOLDER(32), STDLST(1))
C
C
      DATA FLAG/Z1000,Z2000,Z3000,Z4000,Z5000,Z6000,Z0000/
C
C  WHICH SHORT CODE WAS SPECIFIED
      CHOICE =1
      IF(SHRTCD.LT.2.OR.NUMPDS.LT.3) CHOICE =3
      IF(NUMPDS.LT.4) CHOICE=2
      IF(NUMSTU.EQ.0) GO TO 501
      ENDPD=ACSTRT+NUMSKD-1
      AC2=ACSTRT+2
      AC1=ACSTRT+1
      ENO2=ENDPD-2
      ENO1=ENDPD-1
      DO 500 I=1,NUMSTU
      RCODE=FLAG(6)
      IF(STUNUM.GE.0) GO TO 500
      STUNUM=-STUNUM
      POSNUM=DUMMY2(STUNUM)
      IF(POSNUM.LE.0) GO TO 495
      PLACE=MOD(POSNUM,PART)
      IF(PLACE.EQ.0) PLACE=PART
      REQSEC = (POSNUM-1)/PART+1
      IF(INSEC.EQ.REQSEC) GO TO 90
      IF(UP) WRITE(25'INSEC*4-3) STDINF
      UP=.FALSE.
      READ(25'REQSEC*4-3) STDINF
      INSEC= REQSEC
90    CONTINUE
      GO TO (100,200,300), CHOICE
100   MAGIC=0
      DO 110 K=AC2,ENO2
      IF(STDINF(K,PLACE).NE.0) GO TO 420
110   CONTINUE
      IF(STDINF(ACSTRT,PLACE).EQ.0) MAGIC=MAGIC+8
      IF(STDINF(AC1,PLACE).EQ.0) MAGIC=MAGIC+4
      IF(STDINF(ENDPD,PLACE).EQ.0) MAGIC=MAGIC+1
      IF(STDINF(ENO1,PLACE).EQ.0) MAGIC=MAGIC+2
      IF(MAGIC.EQ.0) GO TO 420
      GO TO (420,420,430,420,420,440,450,420,420,420,430,460,460,470,
     1 480), CHOICE
200   MAGIC=0
      DO 210 K=AC1,ENO1
      IF(STDINF(K,PLACE).EQ.0) GO TO 420
210   CONTINUE
      IF(STDINF(ACSTRT,PLACE).EQ.0) MAGIC=MAGIC+2
      IF(STDINF(ENDPD, PLACE).EQ.0) MAGIC=MAGIC+1
      IF(MAGIC.EQ.0) GO TO 420
      GO TO (450,470,480),MAGIC
300   DO 310 K=ACSTRT,ENDPD
      IF(STDINF(K,PLACE).NE.0) GO TO 420
310   CONTINUE
      GO TO 480
420   RCODE=FLAG(6)
      NOPDSD=NOPDSD+NUMSKD
      GO TO 495
430   RCODE=FLAG(2)
      NOPDSD=NOPDSD+2
      A=AC2
      B=ENDPD
      GO TO 490
440   RCODE = FLAG(1)
      NOPDSD=NOPDSD+2
      A=AC1
      B=ENO1
      GO TO 490
450   RCODE= FLAG (5)
      NOPDSD= NOPDSD+1
      A=AC1
      B=ENDPD
      GO TO 490
460   RCODE= FLAG (4)
      NOPDSD=NOPDSD+2
      A=ACSTRT
      B=ENO2
      GO TO 490
470   RCODE= FLAG (3)
```

```
      NOPDSD=NOPDSD+1
      A=ACSTRT
      B=EN01
      GO TO 490
 480  RCODE=FLAG(7)
      A=ACSTRT
      B=ENDPD
      GO TO 490
 490  DO 492 L=A,B
 492  STDINF (L,PLACE)=REQNUM
      UP=.TRUE.
 495  STDLST(I)=STDLST(I)+RCODE
 500  CONTINUE
 501  CONTINUE
      RETURN
      END
```

DRPSKD

```
      SUBROUTINE DRPSKD
C     SUBROUTINE TO DROP A SCHEDULED REQUEST SO THAT IT MAY BE RESCHEDUL
C    ED AT A DIFFERENT TIME
C
      IMPLICIT INTEGER*2 (A-Z)
C
      COMMON /BLOK05/ PART, STDINF(28,256), DUMMY2(2500), INSEC, IN, UP
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531),REQNUM
      COMMON /BLOK30/ TEACRQ(500)
C
C
      LOGICAL*1 IN,UP
C
      INTEGER*4 POSNUM,PART,TCHARY,RUMARY,RQST
C
      DIMENSION ROOMS(10),TCHRS(10),STDLST(2500)
C
C
      EQUIVALENCE (HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     $ (HOLDER(21), NUMPDS), (HOLDER(22), CONMAX), (HOLDER(23), SHRTCD),
     $ (HOLDER(24), START), (HOLDER(25), BETPDS), (HOLDER(26), ANDPDS),
     $ (HOLDER(29), NUMSTU), (HOLDER(30), NUMSKD), (HOLDER(31), ACSTRT),
     $ (HOLDER(32), STDLST(1))
C
C
 1000 FORMAT('0RESCHEDULE OF REQUEST',I5,' FAILED BECAUSE OF PROGRAM '
     1 'ERROR')
C
C
      DO 100 I=1,10
      IF(ROOMS(I).LT.0) GO TO 105
      IF(ROOMS(I).EQ.0) GO TO 120
  100 CONTINUE
      WRITE(6,1000) REQNUM
      RETURN
  105 ROOMS(I)=-ROOMS(I)
      CALL DELETE (RUMARY,ROOMS(I),ACSTRT,NUMSKD)
  120 DO 125 I=1,10
      IF(TCHRS(I).EQ.0) GO TO 130
      CALL DELETE(TCHARY,TCHRS(I),ACSTRT,NUMSKD)
      TEACRQ(TCHRS(I))= TEACRQ(TCHRS(I))+NUMSKD
  125 CONTINUE
  130 CALL DELETE(RQST,REQNUM,ACSTRT,NUMSKD)
      ENDPD=ACSTRT+NUMSKD-1
      DO 200 I=1,NUMSTU
      STUNUM=STDLST(I)
      IF(STUNUM.LE.0) GO TO   190
      POSNUM= DUMMY2(STUNUM)
      IF(POSNUM .LE.0)  GO TO 200
      PLACE=MOD(POSNUM,PART)
      IF(PLACE.EQ.0) PLACE=PART
      REQSEC= (POSNUM-1)/PART+1
      IF(INSEC.EQ.REQSEC)  GO TO 150
      IF(UP)WRITE(25'INSEC*4-3) STDINF
      UP= .FALSE.
      READ (25'REQSEC*4-3) STDINF
      INSEC= REQSEC
  150 CONTINUE
      DO 160 J=ACSTRT,ENDPD
      STDINF (J,PLACE)= 0
  160 CONTINUE
      UP= .TRUE.
      GO TO 200
  190 CONTINUE
      STDLST(I)= -STUNUM
  200 CONTINUE
      RETURN
      END
```

OPENMA

```
      SUBROUTINE OPENMA
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK88/ DAYZ,STARTS,HOUR,SELECT,OUT,KEEPH,RMCNT,COUNT,
     1KEEPN,KEEPZ,DIGIT,KEEPR,PNUM,TOTALP,CHECK,ROOMOK,RTAG
      LOGICAL*1 CHECK(100),RTAG,ROOMOK(10,28)
      LOGICAL*1  TCHOK,SKEDOK
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2500), HOUR(29), SELECT(28),
     1STARTS(840), OUT(7)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1)),(OUT(1),STNUM),(OUT(2),
     4STANUM), (OUT(3),CNUM), (OUT(4),RNUM)
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
C
C        **ALL POSSIBLE START TIMES FORMED HERE**
C
      DO 131 I=1,840
C
C     VARIABLES SET UP HERE.
  131 STARTS(I)=0
      KEEPN= 0
      KEEPZ= 101
      DIGIT= 0
      POINT= 0
      ROOM= 1
C
C     FIND THE BEST LENGTH OF GROUPS OF STARTING PERIODS.
      BEST= TOPEN- LONG+ 1
      IF (BEST.LE.0) BEST= 1
      TEMP= LAST- BEGIN+ 1
  135 NUM=0
      SAVE= POINT
C
C     STARTING AT LAST PERIOD COUNT FORWARD THE NUMBER OF CONSECUTIVE
C     STARTING PERIODS FOR THAT ROOM AND THEN STORE.
      A= LAST+ 1
      DO 140 I= 1, TEMP
      FIRST= A- I
      IF (.NOT.ROOMOK(ROOM,FIRST)) GO TO 137
      NUM= NUM+ 1
      IF (NUM.GT.BEST) NUM= BEST
      POINT= POINT+ 3
      STARTS(POINT-2)= ROOM
      STARTS(POINT-1)= FIRST
      STARTS(POINT)= NUM
      GO TO 140
  137 NUM= 0
  140 CONTINUE
C
C     CHECK TO SEE IF ANY GROUPS OF STARTING PERIODS WERE FORMED.
      IF(POINT.GT.SAVE) GO TO 142
C
C     COUNT THE ROOMS WHERE NO GROUPS WERE FORMED.
      DIGIT= DIGIT+ 1
      GO TO 161
C
C
C
C        **BEST STARTING TIMES LOCATED HERE**
C
C     FOR ROOM NOW IN CONSIDERATION SORT THE VARIOUS STARTING TIMES IN
C     DESCENDING ORDER BY THE NUMBER OF CONSECUTIVE PERIODS FOLLOWING.
  142 A= (POINT-SAVE)/3
      CALL DESCND (STARTS((ROOM-1)*84 + 1),A,THREE)
      BEST2= BEST- LONG
      A= SAVE+3
      DO 160 I=A,POINT,3
      NUM= STARTS(I)
C
C     FIND IF THE LENGTH OF THE GROUP IS ENOUGH FOR ONE CONSECUTIVE
C     GROUP. IF NOT TRANSFER.
      IF (NUM.LE.BEST2) GO TO 161
      ZERO= 0
      FIRST= STARTS(I-1)
      DO 150 K=1,COUNT
C
C     COUNT STUDENTS NOT AVAILABLE.
  150 IF (SKEDOK(DAYZ,K,FIRST,NUM)) ZERO= ZERO+ 1
C
C     IF GROUP IS FULL LENGTH AND ALL STUDENTS ARE AVAILABLE TRANSFER
C     AND SCHEDULE.
      IF (ZERO.LE.0.AND.NUM.GE.BEST) GO TO 164
C
C     CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
      IF ((ZERO.GT.KEEPZ).OR.(ZERO.EQ.KEEPZ.AND.NUM.LE.KEEPN)) GO TO 160
      KEEPZ= ZERO
      KEEPF= FIRST
      KEEPN= NUM
      KEEPR= ROOM
  160 CONTINUE
C
C     POINTERS ADJUSTED AND RETURNED TO CHECK NEXT ROOM, IF ANY.
  161 POINT= ROOM* 84
      ROOM= RCOM+ 1
      IF(ROOM.LE.RMCNT) GO TO 135
C
C     CHECK TO SEE IF ANY OF THE GROUPS ARE ACCEPTABLE LENGTH FOR ONE
C     CONSECUTIVE GROUP. IF NOT TRANSFER
      IF(KEEPZ.GE.101) RETURN
      IF(DIGIT.GE.RMCNT) RETURN
      GO TO 165
  164 KEEPF= FIRST
      KEEPZ= ZERO
      KEEPN= NUM
      KEEPR= ROOM
  165 A= KEEPF+ KEEPN- 1
      PNUM= 0
C
C     SET UP VARIABLES NEEDED LATER IN SCHEDULING STUDENTS.
      DO 170 I= KEEPF,A
      PNUM= PNUM+ 1
  170 SELECT (PNUM)=I
C
C     SET UP VARIABLES NEEDED BY OUTPUT ROUTINES.
      BEGIN= KEEPF
      LAST= KEEPF+ KEEPN- 1
      A= KEEPN+ LONG- 1
C
C     SCHEDULE ROOMS, TEACHERS AND REQUEST NUMBER.
      IF (.NOT.RTAG) GO TO 175
      CALL PLACE (RUMARY, ROOMS(KEEPR), KEEPF, A)
      B= A+KEEPF-1
C
C     SET UP VARIABLES NEEDED FOR OUTPUT ROUTINES.
      DO 173 I=KEEPF,B
  173 SLIST(TSTUD+I)= ROOMS(KEEPR)
  175 DO 180 I=1,10
      IF (TCHRS(I).EQ.0) GO TO 181
  180 CALL PLACE(TCHARY,TCHRS(I),KEEPF,A)
  181 CALL PLACE(RQST,RQNUM,KEEPF,A)
      RETURN
C**********************************************************************
C**********************************************************************
      END
```

OPEN

```
      SUBROUTINE OPEN
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK77/DAYZ,STARTS,STDENT,HOUR,SELECT,RMCNT,COUNT,KEEPZ,
     1DIGIT,PNUM,RTAG,ROOMOK
      LOGICAL*1 STAFF(28), ROOMOK(28,10), CHECK(100), PERIOD(28), RTAG
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2528), HOUR(29), SELECT(28),
     1STARTS(840), STDENT(200)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
 1000 FORMAT(' OPEN REQUEST SCHEDULED, REQUEST NUMBER 'I7)
 1002 FORMAT(' OPEN REQUEST NOT SCHEDULED BECAUSE NUMBER OF STUDENTS = 0
     1,REQUEST NUMBER 'I7)
 1003 FORMAT(' OPEN REQUEST NOT SCHEDULED BECAUSE NO PERIODS OPEN, REQUE
     1ST NUMBER'I7)
C
      PNUM= 0
      CALL OPENA
      IF (COUNT.LE.0) GO TO 440
      CALL OPENB
C     CHECK IF ANY PERIODS ARE AVAILABLE AT ALL. IF NONE TRANSFER.
C
      IF (DIGIT.GE.RMCNT) GO TO 450
      IF (KEEPZ.GE.101) CALL OPENC
      CALL OPEND
C
C     SET UP VARIABLES FOR OUTPUT ROUTINES AND WRITE MESSAGES THEN
C     RETURN.
  400 ACSTRT= 1
      WRITE(6,1000) REQNUM
      IF (RTAG) RETURN
      DO 405 I= 1,28
  405 SLIST(TSTUD+I)= 0
      RETURN
  440 WRITE (6,1002) REQNUM
      ACSTRT= 0
      RETURN
  450 WRITE (6,1003) REQNUM
      ACSTRT= 0
      RETURN
      END
```

OPENB

```
      SUBROUTINE OPENB
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK77/DAYZ,STARTS,STDENT,HOUR,SELECT,RMCNT,COUNT,KEEPZ,
     1DIGIT,PNUM,RTAG,ROOMOK
      LOGICAL*4 STDOK
      LOGICAL*1  TCHOK,SKEDOK
      LOGICAL*1 STAFF(28), ROOMOK(28,10), CHECK(100), PERIOD(28), RTAG
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2528), HOUR(29), SELECT(28),
     1STARTS(840), STDENT(200)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
C
C        **ALL POSSIBLE START TIMES FORMED HERE**
C
      DO 131 I=1,840
C
C     VARIABLES SET UP HERE.
  131 STARTS(I)=0
      KEEPN= 0
      KEEPZ= 101
      DIGIT= 0
      POINT= 0
      ROOM= 1
C
C     FIND THE BEST LENGTH OF GROUPS OF STARTING PERIODS.
      BEST= TOPEN- LONG+ 1
      IF (BEST.LE.0) BEST= 1
      TEMP=LAST-BEGIN+1
  135 NUM=0
      SAVE= POINT
C
C     STARTING AT LAST PERIOD COUNT FORWARD THE NUMBER OF CONSECUTIVE
C     STARTING PERIODS FOR THAT ROOM AND THEN STORE.
      A=LAST+1
      DO 140 I=1,TEMP
      FIRST=A-I
      IF (.NOT.ROOMOK(FIRST,ROOM)) GO TO 137
      NUM= NUM+ 1
      IF (NUM.GT.BEST) NUM= BEST
      POINT= POINT+ 3
      STARTS(POINT-2)= ROOM
      STARTS(POINT-1)= FIRST
      STARTS(POINT)= NUM
      GO TO 140
  137 NUM= 0
  140 CONTINUE
C
C     CHECK TO SEE IF ANY GROUPS OF STARTING PERIODS WERE FORMED.
      IF(POINT.GT.SAVE) GO TO 142
C
C     COUNT THE ROOMS WHERE NO GROUPS WERE FORMED.
      DIGIT= DIGIT+ 1
      GO TO 161
C
C
C
C        **BEST STARTING TIMES LOCATED HERE**
C
C     FOR ROOM NOW IN CONSIDERATION SORT THE VARIOUS STARTING TIMES IN
C     DESCENDING ORDER BY THE NUMBER OF CONSECUTIVE PERIODS FOLLOWING.
  142 A=(POINT-SAVE)/3
      CALL DESCND (STARTS((ROOM-1)*84 + 1),A,THREE)
      BEST2= BEST- LONG
      A= SAVE+3
      DO 160 I=A,POINT,3
      NUM= STARTS(I)
C
C     FIND IF THE LENGTH OF THE GROUP IS ENOUGH FOR ONE CONSECUTIVE
C     GROUP. IF NOT TRANSFER.
      IF (NUM.LE.BEST2) GO TO 161
      ZERO= 0
      FIRST= STARTS(I-1)
      DO 150 K=1,COUNT
C
C     COUNT STUDENTS NOT AVAILABLE.
  150 IF (SKEDOK(DAYZ,K,FIRST,NUM)) ZERO= ZERO+ 1
C
C     IF GROUP IS FULL LENGTH AND ALL STUDENTS ARE AVAILABLE TRANSFER
C     AND SCHEDULE.
      IF (ZERO.LE.0.AND.NUM.GE.BEST) GO TO 164
C
C     CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
      IF ((ZERO.GT.KEEPZ).OR.(ZERO.EQ.KEEPZ.AND.NUM.LE.KEEPN)) GO TO 160
      KEEPZ= ZERO
      KEEPF= FIRST
      KEEPN= NUM
      KEEPR= ROOM
  160 CONTINUE
C
C     POINTERS ADJUSTED AND RETURNED TO CHECK NEXT ROOM, IF ANY.
  161 POINT= ROOM* 84
      ROOM= ROOM+ 1
      IF(ROOM.LE.RMCNT) GO TO 135

      CHECK TO SEE IF ANY OF THE GROUPS ARE ACCEPTABLE LENGTH FOR ONE
      CONSECUTIVE GROUP. IF NOT TRANSFER
      IF(KEEPZ.GE.101) RETURN
      GO TO 165
  164 KEEPF= FIRST
      KEEPZ=ZERO
      KEEPN= NUM
      KEEPR= ROOM
  165 A= KEEPF+ KEEPN- 1
      PNUM= 0

      DO 170 I= KEEPF,A
      PNUM= PNUM+ 1
  170 SELECT (PNUM)=I

      SET UP VARIABLES NEEDED BY OUTPUT ROUTINES.
      BEGIN= KEEPF
      LAST= KEEPF+ KEEPN- 1
      A= KEEPN+ LONG- 1

      SCHEDULE ROOMS, TEACHERS AND REQUEST NUMBER.
      IF (.NOT.RTAG) GO TO 175
      CALL PLACE (RUMARY, ROOMS(KEEPR), KEEPF, A)
      B= A+KEEPF-1

      SET UP VARIABLES NEEDED FOR OUTPUT ROUTINES.
      DO 173 I=KEEPF,B
  173 SLIST(TSTUD+I)= ROOMS(KEEPR)
  175 DO 180 I=1,10
      IF (TCHRS(I).EQ.0) GO TO 181
  180 CALL PLACE(TCHARY,TCHRS(I),KEEPF,A)
  181 CALL PLACE(RQST,RQNUM,KEEPF,A)

      SUM THE NUMBER OF STARTING TIMES AVAILABLE TO EACH STUDENT.
      DO 190 I= 1,COUNT
      A= I*2
      STDENT(A)= SUM(DAYZ(I),KEEPF,KEEPN)
  190 STDENT(A-1)= I
      RETURN
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
      END
```

OPENMB

```
      SUBROUTINE OPENMB
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK88/ DAYZ,STARTS,HOUR,SELECT,OUT,KEEPH,RMCNT,COUNT,
     1KEEPN,KEEPZ,DIGIT,KEEPR,PNUM,TOTALP,CHECK,ROOMOK,RTAG
      LOGICAL*1 STAFF(28), CHECK(100), PERIOD(28), RTAG ,ROOMOK(10,28)
      LOGICAL*1  TCHOK,SKEDOK
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2500), HOUR(29), SELECT(28),
     1STARTS(840), OUT(7)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1)),(OUT(1),STNUM),(OUT(2),
     4STANUM), (OUT(3),CNUM), (OUT(4),RNUM)
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
C
C     IF THE REQUEST MUST BE SCHEDULED IN NON-CONSECUTIVE PERIODS
C     THE TRANSFER IS MADE TO HERE AND THE BEST COMBINATION IS FOUND.
C
C     SET UP VARIABLES.
      NUM= 0
      LAST= 0
      BEGIN=28
      KEEPT= -1
      TOTALP= 0
      A= 280
C
C     SORT ALL OF THE STARTING TIMES, BY NUMBER OF CONSECUTIVE STARTING
C     PERIODS, IN DESCENDING ORDER.
      CALL DESCND (STARTS,A,THREE)
      DO 205 I= 1,28
  205 PERIOD(I)= .TRUE.
      DO 211 I=3,840,3
      IF (STARTS(I).LT.NUM) GO TO 212
      FIRST= STARTS(I-1)
      NUM= STARTS(I)
      TOTAL= 0
      A= FIRST+ NUM- 1
C
C     FIND THE GRAND TOTAL OF STUDENTS AVAILABLE FOR EACH OF THE
C     INDIVIDUAL STARTING TIMES IN THAT GROUP.
      DO 210 J=FIRST,A
  210 TOTAL= TOTAL+ HOUR(J)
```

```
C
C      CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
       IF(TOTAL.LE.KEEPT) GO TO 211
       KEEPF= FIRST
       KEEPN= NUM
       KEEPR= STARTS(I-2)
       KEEPT= TOTAL
  211  CONTINUE
  212  DO 213 I=1,COUNT
  213  CHECK(I)= .FALSE.
       RETURN
C**********************************************************************
C**********************************************************************
       END
```

OPENMY

```
      SUBROUTINE OPENMY
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK88/ DAYZ,STARTS,HOUR,SELECT,OUT,KEEPH,RMCNT,COUNT,
     1KEEPN,KEEPZ,DIGIT,KEEPR,PNUM,TOTALP,CHECK,ROOMOK,RTAG
      LOGICAL*1 STAFF(28), ROOMOK(10,28), CHECK(100), PERIOD(28), RTAG
      LOGICAL*4 STDOK,STDOK2
      LOGICAL*1  TCHOK,SKEDOK
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2500), HOUR(29), SELECT(28),
     1STARTS(840), OUT(7)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1)),(OUT(1),STNUM),(OUT(2),
     4STANUM), (OUT(3),CNUM), (OUT(4),RNUM)
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
 1000 FORMAT(' OPEN MAY REQUEST SCHEDULED, REQUEST NUMBER 'I7)
 1002 FORMAT(' OPEN MAY REQUEST NOT SCHEDULED BECAUSE NUMBER OF STUDENTS
     1= 0, REQUEST NUMBER 'I7)
 1003 FORMAT(' OPEN MAY REQUEST NOT SCHEDULED BECAUSE NO PERIODS WERE '
     $'OPEN,  REQUEST NUMBER ',I7)
 1004 FORMAT(' OPEN MAY REQUEST NOT SCHEDULED BECAUSE ROOMS WERE NOT '
     *'ASSIGNED TO REQUEST,  REQUEST NUMBER ',I7)
C
C
C
C     SET UP VARIABLES FOR USE LATER IN THIS PROGRAM.
      IF(TSTUD.LE.0) GO TO 440
      PNUM= 0
      DO 100 I=1,7
  100 OUT(I)= 0
      CNUM= RLONG
      RLONG= LONG
      A= LAST+ LONG- BEGIN
      IF(TOPEN.GT.A) TOPEN=A
      RTAG= .TRUE.
      RECORD= 0
      RMCNT= 0
      DO 103 I=1,28
  103 SLIST(TSTUD+I)= -1
C
C     FIND WHEN TEACHERS ARE AVAILABLE.
      DO 105 I=BEGIN,LAST
```

```
C
C     FIND WHEN ROOMS ARE AVAILABLE.
  105 STAFF(I)= TCHOK(I,LONG)
      DO 115 I=1,10
      IF (ROOMS(I).EQ.0) GO TO 117
      RMCNT=RMCNT+1
      DO 115 J=BEGIN,LAST
  115 ROOMOK(I,J)= SKEDOK(RUMARY,ROOMS(I),J,LONG).AND.STAFF(J)
  117 IF (RMCNT.EQ.0) GO TO 460
C
C     SET UP STUDENT COUNTER.
  120 COUNT= TSTUD
      IF (COUNT.LE.0) GO TO 440
      IF (COUNT.GT.100) COUNT=100
      DO 125 I= 1,29
  125 HOUR(I)= 0
C
C     USE FIRST 100 STUDENTS TO BUILD AN ARRAY OF STUDENTS BY PERIOD
C     AND COUNT HOW MANY STUDENTS ARE AVAILABLE EACH PERIOD.
      DO 130 I= 1,COUNT
      DAYZ(I)=0
      B=SLIST(I)
      IF(.NOT.STDOK(B,BEGIN,LONG))GO TO 127
      CALL PLACE (DAYZ,I,BEGIN,ONE)
      HOUR(BEGIN)= HOUR(BEGIN)+1
      A= BEGIN+1
      IF (A.GT.LAST) GO TO 130
  127 DO 129 J=A,LAST
      IF(.NOT.STDOK(B,J,LONG))GO TO 129
      CALL PLACE (DAYZ,I,J,ONE)
      HOUR(J)= HOUR(J)+1
  129 CONTINUE
  130 CONTINUE
      CALL OPENMA
      IF(KEEPZ.LT.101) GO TO 131
C
C     CHECK IF ANY PERIODS ARE AVAILABLE AT ALL. IF NONE TRANSFER.
      IF (DIGIT.GE.RMCNT) GO TO 450
      CALL OPENMB
      CALL OPENMC
  131 CALL OPENMD
C
C     SET UP VARIABLES FOR OUTPUT ROUTINES AND WRITE MESSAGES THEN
C     RETURN.
  400 ACSTRT= 1
      WRITE(6,1000) REQNUM
      IF (RTAG) RETURN
      DO 405 I= 1,28
  405 SLIST(TSTUD+I)= 0
      RETURN
  440 WRITE (6,1002) REQNUM
      ACSTRT= 0
      RETURN
  450 WRITE(6,1003) REQNUM
```

```
      ACSTRT= 0
      RETURN
C
C     IF NO ROOMS REQUESTED TRANSFER HERE.
  460 WRITE(6,1004) REQNUM
      ACSTRT=0
      RETURN
C**********************************************************************
C**********************************************************************
      END
```

OPEND

```
      SUBROUTINE OPEND
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK77/DAYZ,STARTS,STDENT,HOUR,SELECT,RMCNT,COUNT,KEEPZ,
     1DIGIT,PNUM,RTAG,ROOMOK
      LOGICAL*4 STDOK
      LOGICAL*1 STAFF(28), ROOMOK(28,10), CHECK(100), PERIOD(28), RTAG
      LOGICAL*1  TCHOK,SKEDOK
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2528), HOUR(29), SELECT(28),
     1STARTS(840), STDENT(200), STUDIN(29)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
      NAMELIST/NAMES/A,B,I,J,KEEPT,KEEPH,KEEPN,LONG,TIME,HOURA,RECORD,
     *COUNT,PNUM,HOUR,STUDIN,STDENT
C
C
C
C        **STUDENT SCHEDULING TAKES PLACE HERE**
C
  300 DO 301 I= 1,29
  301 STUDIN(I)= 0
      RECORD= 0
C
C     STUDENTS SORTED BY NUMBER OF STARTING TIMES AVAILABLE TO EACH.
  302 CALL BLKSRT(STDENT,COUNT,TWO)
      DO 340 I=1,COUNT
      A= STDENT(I*2-1)
      B= RECORD* 100+ A
      IF (STDENT(I*2).LE.0) GO TO 335
      KEEPH = 201
      KEEPN= 3000
      KEEPT= 29
C
C     FIND THE STARTING PERIOD WITH THE LOWEST NUMBER OF STUDENTS STILL
C     AVAILABLE TO BE SCHEDULED AND IF TIE, THEN LOWEST ALREADY
C     SCHEDULED.
      DO 330 J= 1,PNUM
      TIME= SELECT(J)
      HOURA= HOUR(TIME)
```

```
C
C     SEE IF STUDENT IS AVAILABLE THAT PERIOD.
      IF (SKEDOK(DAYZ,A,TIME,ONE))GO TO 330
C
C     CHECK THIS PERIOD AGAINST BEST PERIOD SO FAR.
      IF((HOURA.LT.KEEPH).OR.(HOURA.EQ.KEEPH.AND.STUDIN(TIME).LT.KEEPN))
     1GO TO 320
C
C     IF STARTING TIME IS NOT CHOSEN SUBTRACT ONE FROM NUMBER OF
C     STUDENTS STILL AVAILABLE TO BE SCHEDULED.
      HOUR(TIME)= HOURA- 1
      GO TO 330
C
C     SUBTRACT ONE FROM PREVIOUS BEST PERIOD.
  320 HOUR(KEEPT)= HOUR(KEEPT)- 1
      KEEPH= HOURA
      KEEPN= STUDIN(TIME)
      KEEPT= TIME
  330 CONTINUE
      IF(SLIST(B).LE.0.OR.SLIST(B).GT.2500) GO TO 333
      IF(KEEPT.GT.28) GO TO 333
C
C     SCHEDULE STUDENT.
      CALL STCHNG(SLIST(B), KEEPT, LONG, REQNUM)
C
C     ADD ONE TO THE TOTAL OF REMAINING STUDENTS STILL AVAILABLE TO BE
C     SCHEDULED TO OFFSET SCHEDULING.
      HOUR(KEEPT)= HOUR(KEEPT)+ 1
C
C     ADD ONE TO TOTAL STUDENTS SCHEDULED FOR THAT PERIOD.
      STUDIN(KEEPT)= STUDIN(KEEPT)+ 1
      GO TO 340
  333 WRITE(6,NAMES)
C
C     IF STUDENT NOT SCHEDULED, FLAG SET FOR OUTPUT ROUTINES
  335 SLIST(B)= SLIST(B)+ 24576
  340 CONTINUE
      RECORD= RECORD+ 1
C
C     CHECK TO SEE IF MORE STUDENTS ARE STILL TO BE SCHEDULED.
      B= RECORD* 100
      IF (TSTUD.LE.B) RETURN
      COUNT= TSTUD- B
      IF (COUNT.GT.100) COUNT= 100
C
C     SET UP VARIABLES FOR NEXT ROUND.
      DO 341 I=1,29
  341 HOUR(I)= 0
      DO 350 I= 1,COUNT
      DAYZ(I)= 0
C
C     SET UP STUDENT RAISE AND TOTALS FOR NEXT ROUND.
      A= I+ B
      TIME= SELECT(1)
      IF (.NOT. STDOK(SLIST(A), TIME, LONG)) GO TO 345
      CALL PLACE (DAYZ,I,TIME,ONE)
      HOUR(TIME)= HOUR(TIME)+ 1
  345 IF (PNUM.EQ.1) GO TO 350
       DO 349 J=2,PNUM
      TIME= SELECT(J)
      IF(.NOT.STDOK(SLIST(A),TIME,LONG)) GO TO 349
      CALL PLACE (DAYZ,I,TIME,ONE)
      HOUR(TIME)= HOUR(TIME)+ 1
  349 CONTINUE
  350 CONTINUE
      T1= 1
      T28= 28
      DO 360 I=1,COUNT
      C=I*2
      STDENT(C)= SUM(DAYZ(I),T1,T28)
      STDENT(C-1)= I
  360 CONTINUE
  351 GO TO 302
      END
```

OPENC

```
      SUBROUTINE OPENC
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK77/DAYZ,STARTS,STDENT,HOUR,SELECT,RMCNT,COUNT,KEEPZ,
     1DIGIT,PNUM,RTAG,ROOMOK
      LOGICAL*4 STDOK
      LOGICAL*1  TCHOK,SKEDOK
      LOGICAL*1 STAFF(28), ROOMOK(28,10), CHECK(100), PERIOD(28), RTAG,
     1WITHCK(100)

      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2528), HOUR(29), SELECT(28),
     1STARTS(840), STDENT(200)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1))
      EQUIVALENCE (REQNUM,RQNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
C
C     IF THE REQUEST MUST BE SCHEDULED IN NON-CONSECUTIVE PERIODS
C     THE TRANSFER IS MADE TO HERE AND THE BEST COMBINATION IS FOUND.
C
C     SET UP VARIABLES.
      NUM= 0
      LAST= 0
      BEGIN=28
      KEEPT= -1
      TOTALP= 0
      A= 280

C     SORT ALL OF THE STARTING TIMES, BY NUMBER OF CONSECUTIVE STARTING
C     PERIODS, IN DESCENDING ORDER.
      CALL DESCND (STARTS,A,THREE)
      DO 205 I= 1,28
  205 PERIOD(I)= .TRUE.
      DO 211 I=3,840,3
      IF (STARTS(I).LT.NUM) GO TO 212
      FIRST= STARTS(I-1)
      NUM= STARTS(I)
      TOTAL= 0
      A= FIRST+ NUM- 1
C
```

```
C     FIND THE GRAND TOTAL OF STUDENTS AVAILABLE FOR EACH OF THE
C     INDIVIDUAL STARTING TIMES IN THAT GROUP.
      DO 210 J=FIRST,A
  210 TOTAL= TOTAL+ HOUR(J)
C
C     CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
      IF(TOTAL.LE.KEEPT) GO TO 211
      KEEPF= FIRST
      KEEPN= NUM
      KEEPR= STARTS(I-2)
      KEEPT= TOTAL
  211 CONTINUE
C
C     FIND THE SUM OF STARTING PERIODS EACH STUDENT IS AVAILABLE.
  212 DO 213 I=1,COUNT
      A= 2*I
      STDENT(A)= SUM(DAYZ(I),KEEPF,KEEPN)
      STDENT(A-1)= I
  213 CHECK(I)= .FALSE.
      GO TO 235
C
C
C
C       **FIND NEXT GROUP HERE**
C
  226 KEEPZ= 101
      KEEPN= 0
      KEEPT= 0
      DO 231 I= 3,840,3
      NUM= STARTS(I)
C
C     CHECK TO SEE IF GROUP IS LENGTH NEEDED.
      IF (NUM.GT.LOOK) GO TO 231
C
C     CHECK TO SEE IF IT IS SHORTER THAN THE GROUP JUST EXAMINED.
      IF (NUM.LT.KEEPN) GO TO 232
C
C     CHECK FOR ZERO LENGTH. IF SO TRANSFER TO SCHEDULING.
      IF (NUM.LE.0) RETURN
      FIRST= STARTS(I-1)
C
C     CHECK TO SEE IF ANY PERIODS IN THAT GROUP HAVE ALREADY BEEN
C     SCHEDULED.
      A= FIRST+ NUM+ LONG- 2
      DO 227 J=FIRST,A
      IF (.NOT.PERIOD(J)) GO TO 231
  227 CONTINUE
      TOTAL= 0
      ZERO= 0
      DO 230 J= 1,COUNT
C
C     COUNT HOW MANY STUDENTS ARE AVAILABLE FOR THIS GROUP.
      WITHCK(J)= .NOT.SKEDOK(DAYZ,FIRST,NUM)
      IF (WITHCK(J)) TOTAL= TOTAL+ 1
```

```
C
C     COUNT HOW MANY STUDENTS ARE AVAILABLE FOR THIS GROUP AND
C     PREVIOUSLY SCHEDULED GROUP.
      WITHCK(J)= WITHCK(J).OR.CHECK(J)
  230 IF (.NOT.WITHCK(J)) ZERO= ZERO+ 1
C
C     CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
      IF((ZERO.GT.KEEPZ).OR.(ZERO.EQ.KEEPZ.AND.TOTAL.LE.KEEPT))GO TO 231
      KEEPF= FIRST
      KEEPN= NUM
      KEEPR= STARTS(I-2)
      KEEPZ= ZERO
  231 CONTINUE
C
C     SUM THE PERIODS SO FAR AVAILABLE TO EACH STUDENT.
  232 DO 234 I=1,COUNT
      A= 2*I
  234 STDENT(A)= STDENT(A)+ SUM(DAYZ(I),KEEPF,KEEPN)
C
C     COUNT HOW MANY PERIODS ACTUALLY SCHEDULED.
  235 A= KEEPN+ LONG- 1
      TOTALP= TOTALP+ A
C
C     SCHEDULE ROOMS, TEACHERS AND REQUEST NUMBER.
      IF (.NOT.RTAG) GO TO 238
      ROOM= ROOMS(KEEPR)
      CALL PLACE (RUMARY, ROOM, KEEPF, A)
      B= KEEPF+ A- 1
C
C     SET UP VARIABLES NEEDED FOR OUTPUT ROUTINES.
      DO 236 I= KEEPF,B
  236 SLIST(TSTUD+ I)= ROOM
  238 B= KEEPF+ KEEPN- 1
      IF (KEEPF.LT.BEGIN)BEGIN= KEEPF
      IF (B.GT.LAST) LAST= B
      DO 242 I=1,10
      IF (TCHRS(I).EQ.0) GO TO 243
  242 CALL PLACE (TCHARY, TCHRS(I), KEEPF, A)
  243 CALL PLACE (RQST, RQNUM, KEEPF, A)
C
C     SET UP VARIABLES USED FOR SCHEDULING STUDENTS.
      A= KEEPF+ KEEPN- 1
      DO 245 I= KEEPF,A
      PNUM= PNUM+ 1
  245 SELECT(PNUM)= I
C
C     FIND LENGTH NOW NEEDED TO FILL REST OF REQUEST. TRANSFER IF NONE
C     NEEDED.
      LOOK= TOPEN- TOTALP- LONG+ 1
      IF (LOOK.LE.0) RETURN
      A= KEEPF+ KEEPN+ LONG- 2
C
C     SET UP FLAG TO SAY WHICH PERIODS WERE SCHEDULED.
      DO 247 I= KEEPF,A
  247 PERIOD(I)= .FALSE.
```

```
C
C     SET UP FLAGS THAT TELL IF AN INDIVIDUAL STUDENT IS AVAILABLE
C     OR NOT.
      DO 250 I= 1,COUNT
  250 CHECK(I)= (CHECK(I)).OR.(.NOT.SKEDOK(DAYZ,I,KEEPF,KEEPN))
      GO TO 226
      END
      SUBROUTINE OPENMC
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK88/ DAYZ,STARTS,HOUR,SELECT,OUT,KEEPH,RMCNT,COUNT,
     1KEEPN,KEEPZ,DIGIT,KEEPR,PNUM,TOTALP,CHECK,ROOMOK,RTAG
      LOGICAL*1 STAFF(28), CHECK(100), WITHCK(100), PERIOD(28), RTAG,
     1ROOMOK(10,28)
      LOGICAL*1  TCHOK,SKEDOK
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2500), HOUR(29), SELECT(28),
     1STARTS(840), OUT(7)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1)),(OUT(1),STNUM),(OUT(2),
     4STANUM), (OUT(3),CNUM), (OUT(4),RNUM)
      EQUIVALENCE (REQNUM,RONUM)
      DATA ONE/1/, TWO/2/, THREE/3/
      GO TO 235
C
C
C
C        **FIND NEXT GROUP HERE**
C
  226 KEEPZ= 101
      KEEPN= 0
      KEEPT= 0
      DO 231 I= 3,840,3
      NUM= STARTS(I)
C
C     CHECK TO SEE IF GROUP IS LENGTH NEEDED.
      IF (NUM.GT.LOOK) GO TO 231
C
C     CHECK TO SEE IF IT IS SHORTER THAN THE GROUP JUST EXAMINED.
      IF (NUM.LT.KEEPN) GO TO 235
C
C     CHECK FOR ZERO LENGTH. IF SO TRANSFER TO SCHEDULING.
      IF (NUM.LE.0) RETURN
      FIRST= STARTS(I-1)
C
C     CHECK TO SEE IF ANY PERIODS IN THAT GROUP HAVE ALREADY BEEN
C     SCHEDULED.
      A= FIRST+ NUM+ LONG- 2
      DO 227 J=FIRST,A
      IF (.NOT.PERIOD(J)) GO TO 231
  227 CONTINUE
      TOTAL= 0
      ZERO= 0
      DO 230 J= 1,COUNT
C
C     COUNT HOW MANY STUDENTS ARE AVAILABLE FOR THIS GROUP.
      WITHCK(J)= .NOT.SKEDOK(DAYZ,FIRST,NUM)
      IF (WITHCK(J)) TOTAL= TOTAL+ 1
C
C     COUNT HOW MANY STUDENTS ARE AVAILABLE FOR THIS GROUP AND
C     PREVIOUSLY SCHEDULED GROUP.
      WITHCK(J)= WITHCK(J).OR.CHECK(J)
  123 JE(L)=0
  230 IF (.NOT.WITHCK(J)) ZERO= ZERO+ 1
C
C     CHECK THIS GROUP AGAINST BEST GROUP SO FAR.
      IF((ZERO.GT.KEEPZ).OR.(ZERO.EQ.KEEPZ.AND.TOTAL.LE.KEEPT))GO TO 231
      KEEPF= FIRST
      KEEPN= NUM
      KEEPR= STARTS(I-2)
      KEEPZ= ZERO
  231 CONTINUE
C
C     COUNT HOW MANY PERIODS ACTUALLY SCHEDULED.
  235 A= KEEPN+ LONG- 1
      TOTALP= TOTALP+ A
C
C     SCHEDULE ROOMS, TEACHERS AND REQUEST NUMBER.
      IF (.NOT.RTAG) GO TO 238
      ROOM= ROOMS(KEEPR)
      CALL PLACE (RUMARY, ROOM, KEEPF, A)
      B= KEEPF+ A- 1
C
C     SET UP VARIABLES NEEDED FOR OUTPUT ROUTINES.
      DO 236 I= KEEPF,B
  236 SLIST(TSTUD+ I)= ROOM
  238 B= KEEPF+ KEEPN- 1
      IF (KEEPF.LT.BEGIN)BEGIN= KEEPF
      IF (B.GT.LAST) LAST= B
      DO 242 I=1,10
      IF (TCHRS(I).EQ.0) GO TO 243
  242 CALL PLACE (TCHARY, TCHRS(I), KEEPF, A)
  243 CALL PLACE (RQST, RQNUM, KEEPF, A)
C
C     SET UP VARIABLES USED FOR SCHEDULING STUDENTS.
      A= KEEPF+ KEEPN- 1
      DO 245 I= KEEPF,A
      PNUM= PNUM+ 1
  245 SELECT(PNUM)= I
C
C     FIND LENGTH NOW NEEDED TO FILL REST OF REQUEST. TRANSFER IF NONE
C     NEEDED.
      LOOK= TOPEN- TOTALP- LONG+ 1
      IF (LOOK.LE.0) RETURN
      A= KEEPF+ KEEPN+ LONG- 2
C
```

```
C     SET UP FLAG TO SAY WHICH PERIODS WERE SCHEDULED.
      DO 247 I= KEEPF,A
  247 PERIOD(I)= .FALSE.
C
C     SET UP FLAGS THAT TELL IF AN INDIVIDUAL STUDENT IS AVAILABLE
C     OR NOT.
      DO 250 I= 1,COUNT
  250 CHECK(I)= (CHECK(I)).OR.(.NOT.SKEDOK(DAYZ,I,KEEPF,KEEPN))
      GO TO 226
C*****************************************************************
C*****************************************************************
      END
```

OPENMD

```
      SUBROUTINE OPENMD
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK88/ DAYZ,STARTS,HOUR,SELECT,OUT,KEEPH,RMCNT,COUNT,
     1KEEPN,KEEPZ,DIGIT,KEEPR,PNUM,TOTALP,CHECK,ROOMOK,RTAG
      LOGICAL*4 STDOK,STDOK2
      LOGICAL*1  TCHOK,SKEDOK
      LOGICAL*1 CHECK(100), RTAG, SCAT,ROOMOK(10,28)
      INTEGER*4 DAYZ(100), RUMARY, TCHARY,RQST
      DIMENSION TCHRS(10), ROOMS(10), SLIST(2500), HOUR(29), SELECT(28),
     1STARTS(840), OUT(7)
      EQUIVALENCE(HOLDER(1),ROOMS(1)),(HOLDER(11),TCHRS(1)),
     1(HOLDER(21),LONG),
     1(HOLDER(25),BEGIN), (KEEPH,KEEPF), (STAFF,PERIOD), (HOLDER(26),LAS
     2T), (HOLDER(27),TOPEN), (HOLDER(29),TSTUD), (HOLDER(30),RLONG),
     3(HOLDER(31),ACSTRT),(HOLDER(32),SLIST(1)),(OUT(1),STNUM),(OUT(2),
     4STANUM), (OUT(3),CNUM), (OUT(4),RNUM)
      EQUIVALENCE (REQNUM,RGNUM)
      DATA ONE/1/, TWO/2/, THREE/3/
C
C
C
C        **STUDENT SCHEDULING TAKES PLACE HERE**
C
  300 SCAT= .FALSE.
      IF (PNUM.EQ.1) SCAT= .TRUE.
C
C     CHECK TO SEE IF STUDENTS ARE AVAILABLE THIS PERIOD. IF NOT
C     TRANSFER.
      DO 325 I= 1,TSTUD
      TIME= SELECT(1)
      STNUM= SLIST(I)
      IF (.NOT.STDOK(STNUM,TIME,LONG)) GO TO 310
      ASSIGN 310 TO $WITCH
      GO TO 330
  310 IF (SCAT) GO TO 325
      DO 320 J=2,PNUM
      TIME= SELECT(J)
      IF(.NOT.STDOK(STNUM,TIME,LONG)) GO TO 320
      ASSIGN 320 TO $WITCH
      GO TO 330
  320 CONTINUE
  325 CONTINUE
      RETURN
```

```
C
C     PREPARE VARIABLES TO WRITE OUT MAY RECORD.
  330 STANUM= TIME* 32+ LONG
      RNUM= SLIST(TSTUD+ TIME )
      IF (.NOT.RTAG) RNUM=0
      WRITE (20) OUT
      GO TO $WITCH, (310,320)
C***********************************************************************
C***********************************************************************
      END
```

```
C     SUBROUTINE DELETE
C
C     THIS ROUTINE UNSCHEDULES (TURNS BITS OFF)
C
C
      SUBROUTINE DELETE (ARRAY, SPOT, START, NUM)
C
C
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK01/ TWO(28), WORD
      INTEGER*4 TWO, WORD, ARRAY(1), FULL/Z0FFFFFFF/, ANDD
C
C
C     SET WORD TO ALL BITS ON AND SET END POINT OF BIT TURN OFF
C
      WORD = FULL
      ENDD = START + NUM - 1
C
C     USE DO LOOP TO TURN OFF BITS OF WORD
C
      DO 100 I=START, ENDD
  100 WORD = WORD - TWO (I)
C
C     AND WORD INTO SPECIFIED POSITION OF ARRAY
C
      ARRAY (SPOT) = ANDD (ARRAY(SPOT), WORD)
C
      RETURN
C**********************************************************************
C**********************************************************************
      END
```

ANDD

```
         TITLE 'FORTRAN FUNCTION ANDD--TAKES LOGICAL AND OF TWO WORDS'
ANDD     START 0
*                                  AND IS USED AS THE ENTRY POINT.
         BC     15,10(15)          BRANCH AROUND CONSTANTS IN CALLING
*                                  SEQUENCE
         DC     X'5'
         DC     CL5'ANDD '
*
         ST     2,28(13)           STORE REG. 2 IN SAVE AREA
         L      2,0(1)             PUT ADDRESS OF FIRST ARGUMENT IN REG. 2.
         L      0,0(2)             PUT FIRST ARGUMENT IN REG. 0.
         L      2,4(1)             PUT ADDRESS OF SECOND ARGUMENT IN REG. 2.
         N      0,0(2)             AND FIRST AND SECOND ARGUMENTS -- RESULT
*                                  TO REMAIN IN REGISTER 0.
         L      2,28(13)           RESTORE REGISTER 2.
         MVI    12(13),X'FF'       INDICATE RETURN OF CONTROL.
         BCR    15,14              RETURN TO ADDRESS IN REGISTER 14.
         END
```

PLACE

```
C
C
C     THIS ROUTINE SCHEDULES (TURNS BITS ON)
C
C
      SUBROUTINE PLACE(ARRAY, /SPOT/, /START/, /NUM/)
C
C
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK01/ TWO(28), WORD
      INTEGER*4 TWO, WORD, ARRAY, ORR
      DIMENSION ARRAY(1)
C
C
C     ZERO WORD AND SET END POINT OF BIT TURN ON
C
      WORD = 0
      ENDD = START + NUM - 1
C
C     USE DO LOOP TO TURN ON BITS OF WORD
C
      DO 100 I=START, ENDD
  100 WORD= WORD+ TWO(I)
C
C     OR WORD INTO SPECIFIED POSITION OF ARRAY
C
      ARRAY(SPOT) = ORR(ARRAY(SPOT), WORD)
C
      IF(SPOT.GT.500) WRITE(6,1000) SPOT
 1000 FORMAT(' WARNING, SPOT=',I20)
      RETURN
      END
```

ORR

```
         TITLE 'FORTRAN FUNCTION ORR -- TAKES LOGICAL INCLUSIVE OR '
ORR     START 0
*                                   USED AS ENTRY POINT
         BC     15,8(15)           BRANCH AROUND CONSTANTS IN CALLING
*                                   SEQUENCE.
         DC     X'3'
         DC     CL3'ORR'
*
         ST     2,28(13)           STORE REG. 2 IN SAVE AREA.
         L      2,0(1)             PUT ADDRESS OF FIRST ARGUMENT IN REG. 2.
         L      0,0(2)             PUT FIRST ARGUMENT IN REG. 0.
         L      2,4(1)             PUT ADDRESS OF SECOND ARGUMENT IN REG. 2.
         O      0,0(2)             OR FIRST AND SECOND ARGUMENTS -- RESULT
*                                   TO REMAIN IN REGISTER 0.
         MVI    12(13),X'FF'       INDICATE RETURN OF CONTROL.
         L      2,28(13)           RESTORE REGISTER 2.
         SR 15,15
         BCR    15,14              RETURN TO ADDRESS IN REGISTER 14.
         END
```

```
      SUBROUTINE STUSKD
      IMPLICIT INTEGER*2 (A-Z)
      COMMON /BLOK09/ HOLDER(2531),REQNUM
      COMMON /BLOK05/ PART,STDINF(28,256),DUMMY2(2501),IN,UP
      LOGICAL*4 STDOK,STDOK3
      LOGICAL * 1 NOGO1, NOGO2, IN, UP
      INTEGER * 4 POSNUM, PART
      DIMENSION STDLST(2500), FLAG(7)
      EQUIVALENCE (HOLDER(23), SHRTCD), (HOLDER(29), NUMSTU),
     $ (HOLDER(30),NUMSKD), (HOLDER(31), STRTPD),(HOLDER(32),STDLST(1))
      EQUIVALENCE (NUMSKD, NUMPDS)
      DATA FLAG/Z1000,Z2000,Z3000,Z4000,Z5000,Z6000,Z0000/
      NOGO1 = SHRTCD.LT.2.OR.NUMPDS.LT.3
      NOGO2 = NUMPDS.LT.4
      DO 500 K=1,NUMSTU
      STUNUM= STDLST(K)
      POSNUM= DUMMY2(STUNUM)
      IF (POSNUM.GT.0) GO TO 10
      RCODE= FLAG(6)
      GO TO 500
   10 PLACE = MOD(POSNUM,PART)
      IF(PLACE.EQ.0) PLACE = PART
      A = STRTPD
      B = NUMPDS
      RCODE = FLAG(7)
      IF(STDOK(STUNUM,A,B)) GO TO 100
      RCODE = FLAG(6)
      IF(NOGO1) GO TO 500
      B = NUMPDS - 1
      RCODE = FLAG(3)
      IF(STDOK(STUNUM,A,B))GO TO 100
      A = STRTPD + 1
      RCODE = FLAG(1)
      IF(STDOK(STUNUM,A,B))GO TO 100
      RCODE = FLAG(6)
      IF(NOGO2) GO TO 500
      A = STRTPD
      B = NUMPDS - 2
      RCODE = FLAG(4)
      IF(STDOK(STUNUM,A,B))GO TO 100
      B=NUMPDS
      A = STRTPD + 1
      RCODE = FLAG(5)
      IF(STDOK(STUNUM,A,B))GO TO 100
      A = STRTPD+2
      RCODE = FLAG(2)
      IF(STDOK(STUNUM,A,B))GO TO 100
      RCODE = FLAG(6)
      GO TO 500
  100 ENDIT = A + B - 1
      DO 200 I=A,ENDIT
  200 STDINF(I,PLACE) = REQNUM
      UP = .TRUE.
  500 STDLST(K) = STDLST(K) + RCODE
      RETURN
      END
      BLOCK DATA
```

STDOK
WHATRQ

```
      LOGICAL FUNCTION STDOK*4(STDNUM,STRTPD,NUMPDS)
C
C     TRIPLE ENTRY FUNCTION TO CHECK FOR STUDENT OK TO SCHEDULE
C     AND TO CHECK FOR WHICH REQUEST CAUSES CONFLICT IF NOT OK
C     TO SCHEDULE.
C
C
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK05/ PART, STDINF(28,256), STDLST(2500), INSEC, IN, UP
      COMMON /BLOK07/ CUTOFF, LNSTRT, LNEND
      LOGICAL*1 IN,UP,NOLNCH
      INTEGER*4 PART, POSNUM
C
C
  900 FORMAT (IH0,'STUDENT',I5,' DOES NOT APPEAR IN MASTER FILE',
     $ ' NO SCHEDULE CREATED')
C
C     INSURE THAT PROPER SECTION OF STUDENT INFORMATION FILE IS IN CORE
C     IF NEW SECTION REQUIRED, READ IT IN.  FIRST CHECK TO SEE IF OLD
C     SECTION HAS BEEN UPDATED.  IF IT HAS, WRITE IT OUT.
C
   10 POSNUM = STDLST(STDNUM)
      IF (POSNUM.GT.0) GO TO 15
      WRITE (6,900) STDNUM
      GO TO 80
   15 REQSEC = (POSNUM-1) / PART + 1
      IF (INSEC.EQ.REQSEC) GO TO 20
      IF(UP) WRITE(25'INSEC*4-3) STDINF
      UP = .FALSE.
      READ(25'REQSEC*4-3) STDINF
      INSEC = REQSEC
   20 PLACE = MOD(POSNUM,PART)
      IF (PLACE.EQ.0) PLACE = PART
      ENDPD = STRTPD+NUMPDS-1
      NOLNCH = .FALSE.
C
C     CHECK FOR PERIODS AVAILABLE WITHOUT CONFLICTING WITH LUNCH
C
      DO 30 I=STRTPD,ENDPD
      IF (STDINF(I,PLACE) .NE. 0) GO TO 90
   30 CONTINUE
      IF(STRTPD.GT.LNEND.OR.ENDPD.LT.LNSTRT.OR.NUMPDS.GE.CUTOFF)GO TO 80
      IF(STRTPD .LE. LNSTRT+1) GO TO 50
      ENDLN = STRTPD - 2
      DO 40 I=LNSTRT,ENDLN
      IF(STDINF(I,PLACE)+STDINF(I+1,PLACE).EQ.0) GO TO 80
   40 CONTINUE
   50 IF (ENDPD .GE. LNEND - 1) GO TO 70
      STRTLN = ENDPD+2
      DO 60 I=STRTLN,LNEND
      IF(STDINF(I,PLACE)+STDINF(I-1,PLACE).EQ.0)          GO TO 80
   60 CONTINUE
   70 NOLNCH = .TRUE.
      GO TO 90
   80 STDOK = .TRUE.
      RETURN
   90 STDOK = .FALSE.
      RETURN
C
C     THIS ENTRY POINT DETERMINES THE REQUEST, IF ANY, THAT
C     IS CAUSING THE CONFLICT.  IF MULTIPLE REQUESTS ARE CAUSING THE
C     CONFLICT, OR LUNCH IS IN CONFLICT, THE FUNCTION IS SET TO ZERO,
C     OTHERWISE THE FUNCTION IS SET TO THE NUMBER OF THE REQUEST IN
C     CONFLICT.
C
      ENTRY WHATRQ(/XYZ/)
C
C     XYZ IS AN UNUSED ARGUMENT.  IT IS NECESSARY SO THAT THE COMPILER
C     WILL CONSIDER WHATRQ AS A FUNCTION.
C
      IF (NOLNCH) GO TO 110
      BUG = STDINF(I,PLACE)
      DO 100 K=I, ENDPD
      BUGSVE = BUG
      BUG = STDINF(K,PLACE)
      IF (BUG .EQ. 0) GO TO 100
      IF (BUGSVE.NE.BUG) GO TO 110
  100 CONTINUE
      WHATRQ = STDINF(I,PLACE)
      RETURN
  110 WHATRQ = 0
      RETURN
C***********************************************************************
C***********************************************************************
      END
```

BLKSRT

```
      SUBROUTINE BLKSRT(A,B,C)
C
C     THIS ROUTINE IS USED TO SORT COURSE,TEACHER,AND STUDENT NAMES PRIOR
C     TO THEIR USE IN MAKEING CHANGES TO THESE NAME FILES.
C     'A' IS THE NAME OF A SINGLE DIMENSIONED ARRAY
C     'B' IS THE NUMBER OF ITEMS TO BE SORTED
C     'C' IS THE SIZE OF THE BLOCKS MEASURED IN HALF WORDS
C        NOTE  THE LAST HALF WORD MUST CONTAIN AN INTEGER NUMBER.
C        THE SORT IS MADE ON THIS NUMBER.
C
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      DIMENSION A(1)
C
C
      IF(B.LT.2) RETURN
      N=(B-1)*C
      DO 10 J=C,N,C
      I=J
    9 IF(A(I).LE.A(I+C)) GO TO 10
      L=I-C+1
      DO 11 K=L,I
      TEMP=A(K)
      A(K)=A(K+C)
   11 A(K+C)=TEMP
      I=I-C
      IF(I)10,10,9
   10 CONTINUE
      RETURN
      END
```

DESCND

```
      SUBROUTINE DESCND(A,B,C)
C
C     THIS ROUTINE IS USED TO SORT IN DESCENDING ORDER
C     'A' IS THE NAME OF A SINGLE DIMENSIONED ARRAY
C     'B' IS THE NUMBER OF ITEMS TO BE SORTED
C     'C' IS THE SIZE OF THE BLOCKS MEASURED IN HALF WORDS
C     NOTE  THE LAST WORD OF EACH BLOCK MUST CONTAIN AN INTEGER AS THIS
C     NUMBER IS WHAT THE SORT IS MADE ON.
C
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      DIMENSION A(1)
C
C
      IF(B.LT.2) RETURN
      N=(B-1)*C
      DO 11 J=C,N,C
      I=J
    9 IF(A(I).GE.A(I+C)) GO TO 11
      L=I-C+1
      DO 10 K=L,I
      TEMP=A(K)
      A(K)=A(K+C)
   10 A(K+C)= TEMP
      I=I-C
      IF(I) 11,11,9
   11 CONTINUE
      RETURN
      END
```

```
        TITLE 'FORTRAN FUNCTION SUM -- COUNTS BITS ON IN WORD'
*
*                                    FUNCTION SUM IS CALLED IN FORM
*                                    X = SUM(WORD), OR X = SUM(WORD,STRT,NUM)
*                                    IF ONE ARGUMENT, COUNT OF BITS ON IN
*                                    ENTIRE WORD IS GIVEN.
*                                    IF THREE ARGUMENTS, SUM OF BITS ON
*                                    STARTING WITH BIT STRT+3 THROUGH
*                                    BIT STRT+NUM+2 IS GIVEN.
*                                    FIRST ARGUMENT MUST BE FULL-WORD
*                                    INTEGER FORMAT.
*                                    SECOND AND THIRD ARGUMENTS MUST
*                                    BE HALF-WORD INTEGER FORMAT.
*
SUM     START 0                      SUM IS CONTROL SECTION NAME AND ENTRY
*                                    POINT.
        B     8(15)                  BRANCH AROUND CONSTANTS IN CALLING
        DC    X'3'                   SEQUENCE.
        DC    CL3'SUM'
*
GR0     EQU   0                      RESULT REGISTER
GR1     EQU   1                      ARGUMENT ADDRESS POINTER
GR2     EQU   2                      ADDRESS HOLDER
GR3     EQU   3                      SHIFT COUNT REGISTER
GR4     EQU   4                      BASE REGISTER
*
        STM   GR2,GR4,28(13)         STORE REGISTERS TO BE USED
        BALR  GR4,0
        USING *,GR4
        L     GR2,0(GR1)             PUT ADDRESS OF COUNT-WORD IN GR2
        L     GR0,0(GR2)             PUT COUNT-WORD IN GR0
        CLI   0(GR1),X'00'           IS IT LAST ARGUMENT
        BH    CHECK                  BRANCH IF YES
        L     GR2,4(GR1)             LOAD ADDRESS OF STRT IN GR2
        LA    GR3,3                  PUT 3 IN GR3 SO STRT=1 WILL SHIFT 4
        AH    GR3,0(GR2)             ADD STRT.
        SLL   GR0,0(GR3)             SHIFT COUNT-WORD STRTQO PLACES LEFT
        L     GR2,8(GR1)             PUT ADDRESS OF NUM IN GR2
        LA    GR3,32                 PUT 32 IN GR3 SO NUM=28 WILL SHIFT 4
        SH    GR3,0(GR2)             SUBTRACT NUM
        SRL   GR0,0(GR3)             SHIFT COUNT-WORD RIGHT 32 - NUM PLACES
CHECK   ST    GR0,WORD               PUT COUNT-WORD IN FULL-WORD STORAGE
        TR    WORD,LIST              TRANSLATE ALL 4 BYTES -- RETURNING
*                                    PACKED DECIMAL BYTES WITH COUNT OF
*                                    BITS ON.
        ZAP   DWORD,BYTE0            SUM VALUES OF BYTES
        AP    DWORD,BYTE1
        AP    DWORD,BYTE2
        AP    DWORD,BYTE3
        CVB   GR0,DWORD              PUT SUM IN RESULT REGISTER IN BINARY.
*
```

```
         LM    GR2,GR4,28(13) RESTORE REGISTERS
         MVI   12(13),X'FF'   INDICATE RETURN OF CONTROL
         BR    14
DWORD    DS    D              USED TO HOLD SUM
WORD     DS    0F             USED TO HOLD COUNT-WORD
BYTE0    DS    CL1            FIRST BYTE OF WORD
BYTE1    DS    CL1            SECOND BYTE OF WORD
BYTE2    DS    CL1            THIRD BYTE OF WORD
BYTE3    DS    CL1            FORTH BYTE OF WORD
LIST     DS    0CL256
LIST01   DC    X'0C1C1C2C1C2C2C3C1C2C2C3C2C3C3C4C1C2C2C3C2C3C3C4C2C'
LIST02   DC    X'3C3C4C3C4C4C5C1C2C2C3C2C3C3C4C2C3C3C4C3C4C4C5C2C3C'
LIST03   DC    X'3C4C3C4C4C5C3C4C4C5C4C5C5C6C1C2C2C3C2C3C3C4C2C3C3C'
LIST04   DC    X'4C3C4C4C5C2C3C3C4C3C4C4C5C3C4C4C5C4C5C5C6C2C3C3C4C'
LIST05   DC    X'3C4C4C5C3C4C4C5C4C5C5C6C3C4C4C5C4C5C5C6C4C5C5C6C5C'
LIST06   DC    X'6C6C7C1C2C2C3C2C3C3C4C2C3C3C4C3C4C4C5C2C3C3C4C3C4C'
LIST07   DC    X'4C5C3C4C4C5C4C5C5C6C2C3C3C4C3C4C4C5C3C4C4C5C4C5C5C'
LIST08   DC    X'6C3C4C4C5C4C5C5C6C4C5C5C6C5C6C6C7C2C3C3C4C3C4C4C5C'
LIST09   DC    X'3C4C4C5C4C5C5C6C3C4C4C5C4C5C5C6C4C5C5C6C5C6C6C7C3C'
LIST10   DC    X'4C4C5C4C5C5C6C4C5C5C6C5C6C6C7C4C5C5C6C5C6C6C7C5C6C'
LIST11   DC    X'6C7C6C7C7C8C'
         END
```

TCHOK
TCLNCH

```
      LOGICAL FUNCTION TCHOK*1 (STRTPD,NUMPDS)
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK01/ TWO(28), WORD
      COMMON /BLOK07/ CUTOFF,LNSTRT,LNEND
      COMMON /BLOK08/ TCHARY(500), RUMARY(500), RQST(2500)
      COMMON /BLOK09/ ROOMS(10), TCHRS(10)
      DIMENSION SPOT(10)
      INTEGER*4 TCHARY, RUMARY, RQST
      INTEGER * 4 TWO, WORD
      LOGICAL*1 TCLNCH,CHECK
      EQUIVALENCE(TCHRS(1),SPOT(1))
      CHECK = .FALSE.
      GO TO 10
      ENTRY TCLNCH (STRTPD,NUMPDS)
      CHECK = .TRUE.
   10 TCHOK = .TRUE.
      WORD = 0
      ENDPD = STRTPD + NUMPDS - 1
      DO 100 I=STRTPD, ENDPD
  100 WORD = WORD + TWO(I)
      DO 200 I=1,10
      IF(.NOT.TCHOK.OR.SPOT(I).EQ.0) GO TO 225
  200 TCHOK =ANDD(TCHARY(SPOT(I)),WORD).EQ.0
  225 IF (CHECK) GO TO 250
      RETURN
  250 IF (ENDPD .LT. LNSTRT .OR. STRTPD .GT. LNEND .OR. SPOT(1) .EQ. 0
     $ .OR.NUMPDS.GE.CUTOFF.OR..NOT.TCHOK) RETURN
      TCLNCH = .FALSE.
      IF (STRTPD .LT. LNSTRT + 2) GO TO 300
      ENDLN = STRTPD - 2
      DO 275 I=LNSTRT, ENDLN
      TCLNCH = .TRUE.
      DO 270 K=1,10
      IF ( .NOT. TCLNCH) GO TO 275
      IF (SPOT(K) .EQ. 0) GO TO 272
  270 TCLNCH = ANDD (TCHARY(SPOT(K)), TWO(I)) +
     $ ANDD (TCHARY(SPOT(K)),TWO(I+1)) .EQ. 0
  272 IF (TCLNCH) RETURN
  275 CONTINUE
  300 IF (ENDPD .GT. LNEND - 2) RETURN
      STRTLN = ENDPD + 2
      DO 325 I=STRTLN,LNEND
      TCLNCH = .TRUE.
      DO 320 K=1, 10
      IF (.NOT.TCLNCH) GO TO 325
      IF (SPOT(K) .EQ. 0) GO TO 322
  320 TCLNCH = ANDD (TCHARY(SPOT(K)), TWO(I)) +
     $ ANDD (TCHARY(SPOT(K)), TWO(I-1)) .EQ. 0
  322 IF (TCLNCH) RETURN
  325 CONTINUE
      RETURN
      END
      SUBROUTINE STCHNG(STDNUM, STRTPD, NUMPDS, REQNUM)
C
C     THIS MULTIPLE ENTRY SUBROUTINE SCHEDULES, I.E. PLACES THE
C     REQUEST NUMBER IN THE STUDENT INFORMATION ARRAY.
C     IF THE FIRST ENTRY, STDCHNG, IS TAKEN THE ROUTINE WILL SIMPLY
C     SCHEDULE THE STUDENT.  IF THE SECOND ENTRY, CKCHNG, IS TAKEN
C     THE ROUTINE WILL FIRST CHECK THAT THE STUDENT IS IN THE MASTER
C     FILE, THAT THERE IS NO OTHER REQUEST IN CONFLICT, AND THAT THERE
C     IS NO LUNCH CONFLICT.      IF ANY OF THESE THINGS OCCUR, NOSKED IS
C     SET TO TRUE AND NO SCHEDULE IS MADE.  OTHERWISE, THE STUDENT IS
C     SCHEDULED AS BEFORE.
C
C
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK05/ PART, STDINF(28,256), STDLST(2500), INSEC, IN, UP
      COMMON /BLOK07/ CUTOFF, LNSTRT, LNEND
      LOGICAL*1 IN, UP, NOSKED, CHECK
      INTEGER*4 PART, POSNUM
C
C
      CHECK = .FALSE.
      GO TO 5
      ENTRY CKCHNG (STDNUM, STRTPD, NUMPDS, REQNUM, NOSKED)
      CHECK = .TRUE.
C
C     INSURE THAT STUDENT LIST AND PROPER SECTION OF STUDENT INFORMATION
C     IS IN CORE.
C
    5 IF(IN) GO TO 10
      IN = .TRUE.
      READ (17'7) STDLST
   10 POSNUM = STDLST(STDNUM)
      NOSKED = POSNUM.EQ.0
      IF (NOSKED) RETURN
   15 REQSEC = (POSNUM-1) / PART + 1
      IF (INSEC.EQ.REQSEC) GO TO 20
      IF (UP) WRITE (25'INSEC*4-3) STDINF
      UP = .FALSE.
      READ (25'REQSEC*4-3) STDINF
      INSEC = REQSEC
   20 PLACE = MOD(POSNUM,PART)
      IF (PLACE.EQ. 0) PLACE = PART
      ENDPD = STRTPD + NUMPDS - 1
C     IF SECOND ENTRY, BRANCH TO CHECK ROUTINE--OTHERWISE JUST SCHEDULE.
```

```
C
      IF (CHECK) GO TO 40
   27 DO 30 I=STRTPD, ENDPD
   30 STDINF(I,PLACE) = REQNUM
      UP = .TRUE.
      RETURN
C     CHECK FOR POSSIBLE CONFLICTS, SET FLAG AND RETURN IF ANY FOUND
C     IF NONE FOUND, TRANSFER TO NORMAL SCHEDULE ROUTINE.
C
   40 DO 50 I=STRTPD, ENDPD
      IF (STDINF(I,PLACE) .EQ. 0) GO TO 50
      NOSKED = .TRUE.
      RETURN
   50 CONTINUE
      IF(STRTPD.GT.LNEND.OR.ENDPD.LT.LNSTRT.OR.NUMPDS.GE.CUTOFF)GO TO 27
      IF(STRTPD.LE.LNSTRT+1) GO TO 60
      ENDLN =STRTPD-2
      DO 55 I=LNSTRT,ENDLN
      IF(STDINF(I,PLACE)+STDINF(I+1,PLACE).EQ.0) GO TO 27
   55 CONTINUE
   60 IF(ENDPD.LT.LNEND-1) GO TO 70
   65 NOSKED = .TRUE.
      RETURN
   70 STRTLN = ENDPD + 2
      DO 80 I=STRTLN, LNEND
      IF (STDINF(I,PLACE) + STDINF(I-1,PLACE) .EQ. 0) GO TO 27
   80 CONTINUE
      GO TO 65
      END
```

SKEDOK

```
C     FUNCTION SKEDOK
C
C
C     THIS SUBROUTINE CHECKS FOR PREVIOUS SCHEDULE AT SPECIFIED TIME
C     AND PLACE IN ARRAY
C
C
      LOGICAL FUNCTION SKEDOK*1(ARRAY,/SPOT/,/START/,/NUM/)
C
C
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
      COMMON /BLOK01/ TWO(28), WORD
      INTEGER*4 TWO, WORD, ARRAY(1), ANDD
C
C
C     ZERO WORD AND SET END POINT OF BIT TURN ON
C
      WORD = 0
      ENDD = START + NUM - 1
C
C     USE DO LOOP TO TURN ON BITS OF WORD
C
      DO 100 I=START, ENDD
  100 WORD = WORD + TWO(I)
C
C     CHECK FOR OVERLAP OF BIT TURN ON
C
C
      SKEDOK = ANDD (ARRAY(SPOT), WORD) .EQ. 0
C
      RETURN
      END
```

RGSKED

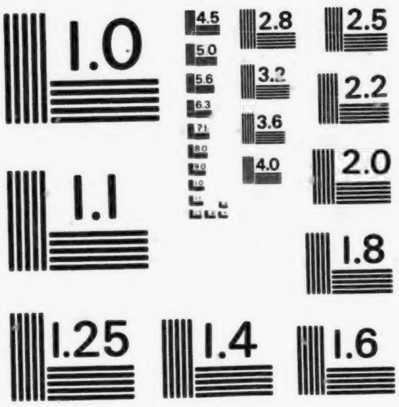
```
      SUBROUTINE RGSKED(*)
C
C     DOUBLE ENTRY SUBROUTINE TO PERFORM REGULAR AND NO PRIORITY
C     SCHEDULING
C
C
C
      IMPLICIT INTEGER * 2 (A-Z)
C
C
      COMMON /BLOK01/ TWO(28), WORD
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      COMMON /BLOK09/ HOLDER(2531), REQNUM
      COMMON /BLOK10/ MATRIX(28)
      COMMON /BLOK30/ TEACRG(500)
C
C
      LOGICAL*4 STDOK,STDOK2
      LOGICAL * 1 SKEDOK, OK, TRYOK(28),NOGO1,NOGO2,TCHOK
      LOGICAL * 1 TCLNCH
      LOGICAL * 1 PTAG
      INTEGER * 4 TCHARY, RUMARY, RQST
      INTEGER * 4 MATRIX, GOBACK, TWO, WORD
C
C
      DIMENSION ROOMS(10), TCHRS(10), RMHOLD(28), NUMCON(28),
     $ STDLST(2500)
C
C
      EQUIVALENCE  (HOLDER,ROOMS), (HOLDER(11), TCHRS),
     1 (HOLDER(21), NUMPDS), (HOLDER(22), CONMAX), (HOLDER(23), SHRTCD),
     2 (HOLDER(24), START), (HOLDER(25), BETPDS), (HOLDER(26), ANDPDS),
     3 (HOLDER(29), NUMSTU), (HOLDER(30), NUMSKD), (HOLDER(31), ACSTRT),
     4 (HOLDER(32), STDLST(1))
  900 FORMAT ('0NO PRIORITY REQUEST',I5,' DENIED--FULL SCHEDULE')
  901 FORMAT ('0REQUEST',I5,' DENIED--PERIODS=',I5)
  902 FORMAT('0REQUEST DENIED--TEACHER OR ROOM CONFLICT--REQUEST',I5)
  903 FORMAT ('0REQUEST DENIED--EXCESSIVE STUDENT CONFLICTS--REQUEST',I5
     $ )
 1000 FORMAT (' REQUEST SCHEDULED, REQUEST NUMBER WAS  ' I10)
 1001 FORMAT(' REQUEST NUMBER',I7,' NOT SCHEDULED BECAUSE OF PROGRAM ERR
     *OR')
C
C
C     SET UP  NUMSKD, ACSTRT, AND BRANCH AROUND SECOND ENTRY.
C
```

```
C -
      NUMSKD = 0
      ACSTRT = 0
      OK= .FALSE.
C THE CARDS TO THE MARKER CARD HAVE BEEN CHANGED TO ACCOMMODATE THE NEW
C     RESCHEDULING ROUTINES
C     GO TO 10
C
C
C     ENTRY FOR NO PRIORITY REQUEST
C
C
C     ENTRY NOPRI (*)
C
C
C     NUMSKD = 0
C     ACSTRT = 0
C     OK= .FALSE.
C
C     SEE IF TEACHER HAS RECEIVED FULL SCHEDULE
C
C     K = TEACRQ(TCHRS(1)) - NUMPDS
C     IF (K.GE.0) GO TO 10
C     WRITE (6,900) REGNUM
C     IF (SHRTCD.EQ.0.OR.SHRTCD.EQ.2.OR.K.LT.(-1).OR.NUMPDS.LT.3)
C    $ RETURN 1
C     NOGO2 = NUMPDS.LT.4
C     GO TO 25
C
C     START OF REG. SCHEDULE OR NO PRIORITY IF TEACHER SCHEDULE NOT
C     FULL
C THIS IS THE MARKER CARD TO MARK THE END OF MODIFICATIONS
   10 NUMSKD = NUMPDS
      NOGO1 = SHRTCD.EQ.0.OR.SHRTCD.EQ.2.OR.NUMPDS.LT.3
      NOGO2 = NUMPDS.LT.4
      ASSIGN 20 TO GOBACK
      GO TO 100
C
C     IF RETURN IS TO HERE,NO SCHEDULE WAS POSSIBLE WITH THAT NUMBER
C     OF PERIODS
C
   20 WRITE (6,901) REGNUM, NUMSKD
      NUMSKD = 0
C
C     NOGO1 IS TRUE IF NO PERIODS SHORTENING IS POSSIBLE
C     NOGO2 IS TRUE IF IT IS NOT POSSIBLE TO SHORTEN TWO PERIODS
C
      IF (NOGO1) RETURN 1
   25 ASSIGN 30 TO GOBACK
      NUMSKD = NUMPDS - 1
      GO TO 100
C
C     IF RETURN IS TO THIS POINT, NO SCHEDULE WAS POSSIBLE WITH PERIODS
C     SHORTENED BY ONE.
```

```
C
   30 WRITE (6,901) REQNUM, NUMSKD
      NUMSKD = 0
      IF (NOGO2) RETURN 1
      ASSIGN 40 TO GO BACK
      NUMSKD = NUMPDS - 2
      GO TO 100
C
C     NO SCHEDULE WITH SCHEDULE SHORTENED BY TWO PERIODS
C
   40 WRITE (6,901) REQNUM, NUMSKD
      NUMSKD = 0
      RETURN 1
C
C     ACTUAL SCHEDULING BEGINS HERE
C
  100 PTAG= .TRUE.
      IF(NUMSKD.LE.6.OR.(ANDPDS-BETPDS).GT.(NUMSKD+2)) PTAG=.FALSE.
      DO 125 I=1,28
  125 TRYOK(I) = PTAG
  150 DO 190 I=BETPDS, ANDPDS
      TRYOK(I) = TRYOK(I).OR.(ANDD(TWO(I), MATRIX(NUMSKD)).NE.0 )
      IF (.NOT. TRYOK(I)) GO TO 190
C
C     CHECK ROOM AVAILABILITY
C
      DO 160 J=1, 10
      IF (ROOMS(J).EQ.0) GO TO 165
      TRYOK(I) = SKEDOK(RUMARY,ROOMS(J),I, NUMSKD)
      IF (TRYOK(I)) GO TO 165
  160 CONTINUE
C
C     PUT ROOM CHOSEN IN RMHOLD(I)
C
  165 IF (.NOT.TRYOK(I)) GO TO 190
      RMHOLD(I) = ROOMS(J)
C
C     CHECK TEACHER AVAILABILITY
C
      TRYOK(I) = TCLNCH(I,NUMSKD)
  190 OK = OK.OR.TRYOK(I)
  200 IF (OK) GO TO 210
C
C     TEACHER OR ROOM CONFLICT
C
      WRITE(6,902) REQNUM
      GO TO GOBACK,(20,30,40)
C
C     CHECK STUDENT AVAILA2ILITY
C
  210 IF(NUMSTU.LE.0) GO TO 305
      DO 230 I=BETPDS, ANDPDS
      IF (.NOT.TRYOK(I)) GO TO 230
      NUMCON(I) = 0
      DO 220 J = 1, NUMSTU
  220 IF (.NOT. STDOK(STDLST(J),I,NUMSKD)) NUMCON(I)=NUMCON(I) + 1
      IF (NUMCON(I).NE. 0) GO TO 230
C
C     SCHEDULE WITH ZERO CONFLICTS
C
      SKDPNT = I
      GO TO 250
  230 CONTINUE
C
C     FIND LEAST NUMBER OF CONFLICTS
C
      SMALST = 32000
      DO 240 I=BETPDS,ANDPDS
      IF (.NOT.TRYOK(I)) GO TO 240
      IF (NUMCON(I).GE. SMALST) GO TO 240
      SMALST = NUMCON(I)
      SKDPNT = I
  240 CONTINUE
C
C     SEE IF LEAST NUMBER OF CONFLICTS IS ACCEPTABLE
C
  250 IF (NUMCON(SKDPNT).LE.CONMAX) GO TO 260
C
C     IT  ISN'T
C
      WRITE (6,903) REQNUM
      GO TO GOBACK,(20,30,40)
C
C     IT IS OK SO SCHEDULE
C
  260 ACSTRT=SKDPNT
      CALL STUSKD
  265 IF (RMHOLD(SKDPNT).EQ.0) GO TO 270
      CALL PLACE(RUMARY,RMHOLD(SKDPNT), SKDPNT,NUMSKD)
  270 DO 280 I=1, 10
      IF (TCHRS(I).EQ.0) GO TO 290
  280 CALL PLACE (TCHARY, TCHRS(I), SKDPNT, NUMSKD)
  290 CALL PLACE (RQST, REQNUM, SKDPNT, NUMSKD)
      DO 300 I=1,10
      IF(RMHOLD(SKDPNT).NE.ROOMS(I)) GO TO 300
      ROOMS(I) = - ROOMS(I)
      WRITE (6,1000) REQNUM
      RETURN 1
  300 CONTINUE
      WRITE(6,1001) REQNUM
      ACSTRT= 0
      RETURN 1
  305 DO 310 I= BETPDS,ANDPDS
      IF(.NOT.TRYOK(I)) GO TO 310
```

4 OF 4

ED
037097


1.0
4.5
5.0
5.6
6.3
7.1
8.0
9.0
10
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
      ACSTRT= I
      SKDPNT= I
      GO TO 265
  310 CONTINUE
      WRITE(6,1001) REQNUM
      ACSTRT= 0
      RETURN 1
      END
```

4

```
C
C     BLOCK DATA DECK DATA01
C
      BLOCK DATA
      COMMON /BLOK01/ TWO(28), WORD
      INTEGER*4 TWO, WORD
      DATA TWO / Z8000000, Z4000000, Z2000000, Z1000000, Z800000,
     1 Z400000, Z200000, Z100000, Z80000, Z40000, Z20000, Z10000,
     2 Z8000, Z4000, Z2000, Z1000, Z800, Z400, Z200, Z100, Z80, Z40,
     3 Z20, Z10, Z8, Z4, Z2, Z1/
C
C     EACH WORD OF THE TWO ARRAY HAS ONLY A SINGLE BIT ON
C     TWO(1) HAS BIT 4 ON, TWO(2) HAS BIT 5 ON, UP TO TWO(28) WHICH
C     HAS BIT 31 ON.
C
      END
```

```
C      DECK TO HOLD BLOK05 AND BLOK07 WITH INITIALIZATION
C
       BLOCK DATA
       IMPLICIT INTEGER*2 (A-Z), INTEGER*4 ($)
       COMMON /BLOK05/ PART, STDINF(28,256), STDLST(2500), INSEC, IN, UP
       COMMON /BLOK07/ CUTOFF, LNSTRT, LNEND
       LOGICAL * 1 IN, UP
       INTEGER *4 PART
       DATA INSEC/0/, IN/F/, PART/256/, UP/F/
       DATA CUTOFF/12/, LNSTRT/13/, LNEND/18/
       END
```

```
      BLOCK DATA
      COMMON /BLOK08/ TCHARY(500),RUMARY(500),RQST(2500)
      INTEGER*4  RUMARY, TCHARY,RQST
      DATA TCHARY/500*0/, RUMARY/500*0/, RQST/2500*0/
      END
C
```

```
C
C     BLOCK DATA SUBPROGRAM TO INITIALIZE STARTING MATRIX
C
C
      COMMON /BLOK10/ MATRIX(28)
C
C
      INTEGER * 4 MATRIX
C
C
      DATA MATRIX /ZFFFFFFFF, Z0AAAAAAA, Z09224908, Z08220888,
     $             Z08201010, Z08202020, Z00200040, Z00200080,
     $             Z08000100, Z08000200, Z08000400, Z08000800,
     $             Z08001000, Z08002000, Z08004000, Z08008000,
     $             Z08010000, Z08020000, Z08040000, Z08080000,
     $             Z08100000, Z08200000, Z08400000, Z08800000,
     $             Z09000000, Z0A000000, Z0C000000, Z08000000/
      END
```

Job Step 10

WRYTOUT MAIN

```
C         WRITE MASTER
C         THERE ARE FOUR JOB STEPS IN THE WRITEOUT SYSTEM
C         THIS IS THE SECOND STEP
C         IN THIS STEP THERE ARE ELEVEN SUBROUTINES AND A MAIN PROGRAM
C         THERE ARE ALSO THREE BLOCK COMMON SUBROUTINES
C         THE MAIN ROUTINE HAS IN IT ALL THE COMMON BLOCKS AND ALL
C         THE DEFINE FILE STATEMENTS
C
C         THE SUBROUTINES ARE.........
C
C         NAME      USAGE
C
C         RUMRYT    WRITES OUT ROOM SCHEDULES
C         ANDD      ASEMBLY FUNCTION USED TO CHECK BIT PATTERN IN RUMARY
C         CRSRYT    HANDLS WRITE OUT OF NON SCHEDULED, NO PRIORITY,AND
C                   REGULAR REQUESTS
C         RUMOUT    WRITES OUT RECORDS OF RUMINF AND MAYINF
C         GRPRYT    HANDLS WRITE OUT OF GROUP,OPEN,MAY AND OPEN-MAY REQUESTS
C         SNAM      USED TO RETRIEVE STUDENT NAMES FROM FILE
C         WHEN      A FUNCTION WHICH FINDS WHIC PERIOD A STUDENT WAS
C                   SCHEDULED IN BY A REQUEST
C         STURYT    HANDLS STUDENT SCHEDULES
C         WHICH     ASEMBLES THE SCHEDULE FOR A PARTICULAR STUDENT
C         WHATRM    FINDS THE ROOM NUMBERS USED BY A REQUEST
C         BIGSRT    SORTS FULL WORD ARRAYS IN ASCENDING ORDER
C
C         THE BLOK COMMON SUBPROGRAMS ARE....
C         BLOK01,BLOK05,AND BLOK11
C
C
C
C
C                                     ************************
C                                     * VARIABLE DICTIONARY *
C                                     ************************
C
C        VARIABLE  USAGE                          COMMENTS
C
C        RUMARY    HOLDS THE ROOMS SCHEDULE IN A BIT PATTERN
C********* RUMARY IS IN BLOK88
C        RUMINF    HOLDS RECORDS CONTAINING ROOMS USED BY REGULAR
C                  AND NO PRIORITY REQUESTS, IT ALSO HAS POINTERS TO
C                  MAYINF
C        MAYINF    HOLDS RECORDS OF THE FILE CONTAINING THE ROOMS
C                  USED BY GROUP,MAY,OPEN,OPENMAY REQUESTS
C        IN        IS A POINTER SHOWING WHICH RECORD OF RUMINF IS IN CORE
C        IN2       IS A POINTER SHOWING WHICH RECORD OF MAYINF IS IN CORE
C        CRSE      IS AN ARRAY HOLDING THE COURSE NUMBERS OF EACH REQUEST
C******** RUMINF--CRSE ARE IN BLOK13
C        CRSS      AN ARRAY USED BY STURYT TO HOLD COURSE NUMBERS
C        RMSS      AN ARRAY USED BY STURYT TO HOLD ROOM NUMBERS
C        STUNUM    USED BY STURYT TO PASS STUDENT NUMBERS TO OTHER ROUTINES
```

```
C******** CRSS,RMSS,STUNUM ARE IN BLOK12
C        TYPE      AN ARRAY HOLDING ALPHANUMERIC DATA  EG. GROUP REQUEST
C        NOROOM    CONTAINS 'NO ROOM'
C        DATE     AAN ARRAY CONTAINING THE DATE THE SCHEDULE
C                  IS TO BE IMPLIMENTED
C        RQDATA    AN ARRAY USED TO HOLD RECORDS FROM THE REQUEST FILE
C        TIME      AN ARRAY CONTAINING THE START TIME FOR EACH PERIOD
C        TYME      AN ARRAY CONTAINING THE END TIME FOR EACH PERIOD
C******** TYPE--TYME ARE IN BLOK11
C        NAMESS    AN ARRAY USED TO HOLD RECORDS FROM THE NAME FILES
C        SAM       AN ARRAY USED BY SNAM TO RETURN A STUDENTS NAME
C        AV11-AV25 ARE THE ASSOCIATED VARIABLES OF DIRECT ACCESS FILES
C******** NAMESS--AV25 ARE IN BLOK10
C        PART      CONTAINS THE NUMBER OF STUDENT SCHEDULES PER RECORD
C                  IN THE STUDENT SCHEDULE FILE
C        STDINF    AN ARRAY USED TO HOLD RECORDS OF THE STUDENT
C                  SCHEDULE FILE
C        STDLST    AN ARRAY HOLDING POINTERS INTO STDINF
C        INSEC     SHOWS WHICH RECORD OF THE STUDENT SCHEDULE FILE
C                  IS IN CORE
C        INN       SHOWS THAT A RECORD IS IN STDINF
C        UP        NOT USED BY THE WRITE OUT ROUTINES
C******** PART--UP ARE IN BLOK05
C        TWO       AN ARRAY WITH ONLY ONE BIT TURNED ON IN EACH WORD
C                  IS USED BY RUMRYT TO OBTAIN A ROOMS SCHEDULE
C        WORD      NOT USED BY THE WRITE OUT ROUTINES
C********* TWO AND WORD ARE IN BLOK01
C        CRSNO     CONTAINS THE CURRENT COURSE NUMBER OR A POINTER
C        B         CONTAINS THE CURRENT REQUEST NUMBER
C
C
      IMPLICIT INTEGER*2(A-Z), INTEGER*4($)
C
      COMMON /BLOK88/ RUMARY(500)
      COMMON /BLOK13/ RUMINF(1250), MAYINF(28,11),IN,IN2,CRSE(2500)
      COMMON/BLOK12/ CRSS(2,28), RMSS(2,28),STUNUM
      COMMON /BLOK11/ TYPE(3,7),NOROOM,DATE(2),RQDATA(81),TIME(3,30),
     1TYME(3,30)
      COMMON /BLOK10/NAMESS(141),SAM(3),AV11,AV12,AV13,AV14,AV15,AV16,
     1 AV17,AV21,AV23,AV25
      COMMON /BLOK05/ PART,STDINF(28,256),STDLST(2500),INSEC,INN,UP
      COMMON/BLOK01/ TWO(28), WORD
      COMMON/BLOKAZ/ LOKFIL (1250),R
      LOGICAL * 1 INN,UP
      INTEGER*4 RUMARY
      INTEGER*4 PART
      INTEGER*4 TWO, WORD
      REAL*8 TYPE,NOROOM,DATE
      REAL*8 NAMESS, SAM
C
C
 1001 FORMAT (2A8)
C
C
```

```
C       FILE11 IS THE TEACHER NAME FILE, FILE 12 IS THE ROOM NAME FILE
C       FILE 13 IS THE STUDENT NAME FILE, FILE 14 IS THE TEACHER NAMES FILE
C      **************************************************************
C      * FILE 15 IS THE COURSE FILE POINTERS FILE                   *
C      * IN WRITE OUT ONLY THE LAST 24 HALF WORDS ARE USED          *
C      * THEY  ARE POINTERS SHOWING WHICH STUDENT IS IN WHICH CLASS *
C      **************************************************************
C       FILE 16 IS THE REQUEST FILE
C       FILE 17 HAS CRSE,STDLST,RUMINF
C       FILE 23 IS THE MAYINF FILE, FILE 25 IS THE STUDENT SCHEDULE FILE
C
C
      DEFINE FILE 11(8,250,U,AV11)
      DEFINE FILE 12(4,250,U,AV12)
      DEFINE FILE 13(54,282,U,AV13)
      DEFINE FILE 15(1,2050,L,AV15)
      DEFINE FILE 14(11,282,U,AV14)
      DEFINE FILE 16(600,162,L,AV16)
      DEFINE FILE 17(8,625,U,AV17)
      DEFINE FILE 23(10,154,U,AV23)
      DEFINE FILE 25(40,896,U,AV25)
C
C
C                                   ********************
C                                   * READ IN THE DATE *
C                                   ********************
C
      READ(8,1001) DATE
C
C                                   ***********************************
C                                   * READ IN THE ROOM SCHEDULE ARRAY *
C                                   ***********************************
C
      READ(24) RUMARY
C
C                                   *****************************
C                                   * CALL THE ROUTINE TO PRINT *
C                                   * OUT THE ROOM SCHEDULE     *
C                                   *****************************
C
      CALL RUMRYT
C
C                                   ********************************
C                                   * READ IN THE STUDENT POINTERS *
C                                   * AND SET THE INDICATOR        *
C                                   ********************************
C
C
      READ(17'7) STDLST
      INN=.TRUE.
C
C                                   ************************************
C                                   * READ IN THE COURSE NUMBERS ARRAY *
C                                   ************************************
C
C
```

```
      READ(17'3) CRSE
C                                             ***********************************
C                                             * COURSE ROLLS WRITEOUT SECTION *
C                                             ***********************************
C
C        SET CRSNO FOR CRSRYT TO START PROCESSING
      CRSNO=1
C
C        BRANCH TO CALL CRSRYT
      GO TO 2
C
C       ****************************************************************
C       * WHEN CRSNO IS PASSED FROM CRSRYT TO GRPRYT IT WILL CONTAIN *
C       * THE ACTUAL NUMBER OF THE COURSE BEEING WORKED ON           *
C       * WHEN GOING FROM GRPRYT TO CRSRYT IT SERVES AS AN INTICATOR *
C       * TO CRSRYT AS TO WHAT SHOULD BE DONE                        *
C       ****************************************************************
C
  888 CRSNO=0
    2 CALL CRSRYT(+999,CRSNO,B)
      CALL GRPRYT(+888,CRSNO,B)
C                                             ***********************************
C                                             * CALL THE ROUTINE TO PRINT OUT *
C                                             * THE STUDENT SCHEDULES          *
C                                             ***********************************
  999 CALL STURYT
      CALL EXIT
      END
      SUBROUTINE  RUMRYT
C        THIS SUBROUTINE WRITES OUT THE ROOM SCHEDULES.
C        IT IS CALLED ONCE FROM THE MAIN PROGRAM
C        IT CALLS THE ASEMBLY FUNCTION ANDD
C
C        VARIABLE  USAGE
C
C        XRUM         IS AN ARRAY TO HOLD ONE ROOMS SCHEDULE
C        X            PLACED IN XRUM DENOTES THAT THE ROOM WAS SCHEDULED
C        O            PLACED IN XRUM DENOTES THAT THE ROOM WAS NOT SCHEDULED
C        NONAME       USED TO CHECK FOR VALID ROOM NAMES
C        RC           IS A COUNTER , COUNTS THE NUMER OF SCHEDULES ON A PAGE
C        NAMFI        AN ARRAY TO HOLD ROOM NAME FILE RECORDS
C                     IT IS EQUIVALENCED TO NAMESS
C        I            IS THE MAIN DO LOOP PARAMETER
C        J            IS AN ADJUSTMENT PARAMETER FOR A NAME IN A RECORD
C        K            INSIDE DO LOOP PARAMETER
C
      IMPLICIT INTEGER*2 (A-Z), INTEGER*4(S)
C
C       FOR A DESCRIPTION OF THE VARIABLES IN COMMON SEE THE MAIN PROGRAM
C
      COMMON/BLOK01/ TWO(28), WORD
      COMMON /BLOK88/ RUMARY(500)
      COMMON /BLOK10/NAMESS(141),SAM(3),AV11,AV12,AV13,AV14,AV15,AV16,
     1 AV17,AV21,AV23,AV25
      COMMON /BLOK11/ TYPE(3,7),NOROOM,DATE(2),RODATA(81),TIME(3,30),
     1 TYME(3,30)
      INTEGER*4 RUMARY
      INTEGER*4 TWO, WORD
      REAL*8 TYPE,NOROOM,DATE
      REAL*8 NAMESS,SAM
      REAL*8 NAMFI(125), NONAME
      EQUIVALENCE(NAMFI(1),NAMESS(1))
      DIMENSION  XRUM(28)
      DATA X,O, NONAME /' X',' O','NO NAME '/
C
C
 1000 FORMAT ('1',T5,'ROOM SCHEDULE FOR ',2A8,/T5,'ROOM NAME',I3,27I4 )
 2000 FORMAT ('0',T5,A8,28(2X,A2))

C
C
C       FIND THE FIRST ROOM NAME FILE RECORD
      FIND(12'1)
C
C       SET THE ASSOCIATED VARIABLE ACORDINGLY
      AV12=1
C
C       INITIALIZE THE ROOM COUNT TO ZERO
      RC=0
C
```

```
C
C                                    ******************************
C                                    * PRINT OUT THE PAGE HEADING *
C                                    ******************************
      WRITE(6,1000) DATE,(K,K=1,28)
C
C
C
C                                    **********************************
C                                    * READ IN FIRST ROOM NAME RECORD *
C                                    **********************************
C
      READ(12'AV12) NAMFI
C
C
C                                    *************************************
C                                    * ALL FURTHER OPERATIONS ARE INSIDE *
C                                    * THE DO LOOP                       *
C                                    *************************************
C
   10 DO 200 I=1,500
C
C        CALCULATE THE ADJUSTMENT PARAMETER
      J= I-(125*(AV12-2))
C
C        CHECK ..IF THERE IS NO ROOM NAME BRANCH TO PICK UP THE NEXT NUMBER
      IF(NAMFI(J).EQ.NONAME) GO TO 199
C
C                                    *************************************
C                                    * ASEMBLE THE SCHEDULE FOR ONE ROOM *
C                                    *************************************
C
      DO 100 K=1,28
C
C        SET THE INDICATOR TO SHOW THE ROOM WAS SCHEDULED
      XRUM(K)=X
C
C        CHANGE IT IF NEEDED
  100 IF(ANDD(RUMARY(I),TWO(K)).EQ.0) XRUM(K)=0
C
C                                    ******************************
C                                    * WRITE OUT A ROOMS SCHEDULE *
C                                    ******************************
C
      WRITE(6,2000) NAMFI(J),XRUM
C
C        INCREMENT THE ROOM CONUT
      RC=RC+1
C
C        CHECK THE ROOM COUNT. IF LESS THAN 25 BRANCH TO GET THE NEXT NUMBER
      IF(RC.NE.25) GO TO 199
C
C                                    ******************************
C                                    * PRINT OUT THE PAGE HEADING *
C                                    ******************************
C
```

```
      WRITE(6,1000) DATE,(K,K=1,28)
C
C         RESET THE ROOM COUNT
      RC=0
C
C         CHECK THE NEXT ROOM NUMBER. IF ITS NAME IS NOT PRESENTLY IN
C         CORE, READ IN THE NEXT RECORD.
C
  199 IF(AV12-2.EQ.I/125) GO TO 200
C
C         CHECK TO SEE IF THE ONE JUST COMPLETED WAS THE LAST ONE
      IF(I.EQ.500) RETURN
C
C                                              ****************************************
C                                              * READ IN THE NEXT ROOM NAME FILE RECORD
C                                              ****************************************
C
      READ(12'AV12)NAMFI
  200 CONTINUE
      RETURN
      END
      SUBROUTINE CRSRYT(*,CRSNO,B)
C
C     THIS ROUTINE WRITES OUT THE COURSE ROLLS FOR REGULAR, NO-PRIORITY
C     AND NON SCHEDULED REQUESTS.
C     WHEN ANOTHER TYPE REQUEST IS ENCOUNTERED GRPRYT IS CALLED.
C     THIS ROUTINE ALSO PUTS OUT INFORMATION  FOR THE TEACHER ROLLS AND
C     ROOM INFORMATION FOR THE STUDENT SCHEDULES.
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON /BLOK13/ RUMINF(1250), MAYINF(28,11),IN,IN2,CRSE(2500)
      COMMON /BLOK11/ TYPE(3,7),NOROOM,DATE(2),RQDATA(81),TIME(3,30),
     1 TYME(3,30)
      COMMON /BLOK10/NAMESS(141),SAM(3),AV11,AV12,AV13,AV14,AV15,AV16,
     1 AV17,AV21,AV23,AV25
      COMMON/BLOKAZ/ LOKFIL (1250),R
      REAL*8 ROOM,COURSE
      REAL*8 TYPE,NOROOM,DATE
      REAL*8 NAMESS,SAM
      REAL*8 ROOMS(10),STAFF(3)
      REAL*8 NAMES(125),TECH(15),TECHER(3,5),SNAME(15),SNAMES(3,5)
      DIMENSION STIME(3),STYME(3),ASEMBL(10)
      EQUIVALENCE(ROOM,ASEMBL(3)),(COURSE,ASEMBL(7)),(PERIOD,ASEMBL(1)),
     1(PDS,ASEMBL(2))
      EQUIVALENCE (NUMPDS,RQDATA(30)),(ACSTRT,RQDATA(31))
      EQUIVALENCE(NAMES(1),NAMESS(1)),(TECH(1),TECHER(1)),
     1 (SNAME(1),SNAMES(1,1))
      DATA  ZERO /0/
      DATA STAFF /'STAFF   ','        ','        '/
C
C
 1000 FORMAT('2','CLASS ROLL COURSE ',A8,T38,2A8,T61,'TIME  ',6A2,T88,
     1'ROOM - ',A8,T106,3A8,/' ')
 2000 FORMAT(' ',3A8,4(2X,3A8))
 2001 FORMAT(/,'0')
 3000 FORMAT(' ',3(3A8,3X,6A2,4X))
C
C     FIND THE INPUT FILE AND PREPARE THE OUTPUT FILE
C
      IF(CRSNO) 3,999,3
    3 FIND(17'1)
      AV17=1
      K=0
    1 B=0
C
C     READ IN THE POINTERS FILE
      READ(17'AV17) LOKFIL
    2 B=B+1
C
C     CHECK FOR LAST REQUEST
      IF(LOKFIL(B).EQ.0) GO TO 988
```

```
C
C     FIND REQUEST DATA RECORD FOR THE REQUEST BEING WORKED ON
      AV16 =LOKFIL(B)
      FIND(16'AV16)
C
C     BEGIN INITIALIZATION FOR STANDARD REQUEST
      SV=3
      AV13=0
C
C     READ IN REQUEST DATA RECORD
      READ(16'AV16) RQDATA
C     OBTAIN COURSE NUMBER FOR THE REQUEST
      R= R+1
      CRSNO=CRSE(R)
C
C     CHECK FOR NOT SCHEDULED REQUEST
      IF(ACSTRT.EQ.0) GO TO 11
C
C     CHECK FOR GROUP,OPEN,MAY,OPEN-MAY TYPE REQUEST
      IF(RQDATA(28).GT.1) RETURN
C
C     LOCATE THE ROOM NUMBER TO BE USED ON THIS REQUEST.
C     IT IS A NEGATIVE NUMBER OR THE FIRST ZERO ENCOUNTERED
      DO 10 I=1,10
      IF(RQDATA(I)) 12,11,10
   10 CONTINUE
   11 ROOM= NO ROOM
      CALL  RUMOUT(ZERO)
      GO TO 13
C
C     READ IN THE ROOM NAME
   12 A=-RQDATA(I)
      AV12=((A-1)/125)+1
      READ(12'AV12) NAMES
      CALL RUMOUT(A)
      ROOM= NAMES(A-(AV12-2)*125)
C
C     LOCATE THE COURSE NAME
   13 AV11=((CRSNO-1)/125)+1
      FIND(11'AV11)
      IF(ACSTRT.NE.0) GO TO 14
C
C     INITIALIZE FOR NOT SCHEDULED REQUEST
      START= 29
      END  =29
      MSG=7
      SV=5
      GO TO 15
C
C     COMPLETE INITIALIZATION FOR A REGULAR REQUEST
   14 START= ACSTRT
      END = NUMPDS+ START-1
      MSG=RQDATA(28)+1
      PDS= NUMPDS*8 +MSG
      PERIOD = ACSTRT
```

```
C
C     OBTAIN THE COURSE NAME
   15 READ(11'AV11) NAMES
      COURSE= NAMES(CRSNO-(AV11-2)*125)
C
C     WRITE OUT THE HEADING
      WRITE(6,1000) COURSE,DATE(1),DATE(2),(TIME(I,START),I=1,3),
     1(TYME(I,END),I=1,3),ROOM,(TYPE( I  ,MSG),I=1,3)
C
C     INITIALIZEFOR WRITE OUT OF TEACHERS IN THE REQUEST
      TT=0
      K=0
      AV14=0
C
C     LOCATE THE TEACHER NUMBER(S). ALL ARE POSATIVE
      DO 20 I=1,10
      NUM= RQDATA(10+I)
      IF(NUM) 20,21,22
C
C     COUNT OF TEACHERS
   22 TT=TT+1
C
C     IF REQUEST IS NOT SCHEDULED SKIP WRITE OUT OF TEACHER INFORMATION
C
C     WRITE OUT OF TEACHER INFORMATION
      IF(AV14  .EQ.((NUM-1)/47)+2) GO TO 16
      AV14=((NUM-1)/47)+1
      READ(14'AV14) NAMESS
   16 IF(MSG.EQ.7) GO TO 18
      PERIOD= NUM*32+PERIOD
      WRITE(21) ASEMBL
C
C     READ IN OF THE TEACHER NAME
 18   K=TT
      F=(NUM-(AV14-2)*47)*3-3
      DO 19 H=1,3
   19 TECHER(H,K)= NAMESS(F+H)
      IF(TT.LT.5.AND.I.LT.10) GO TO 20
C
C     WRITE OUT OF THE TEACHER NAMES FIVE AT A TIME
      WRITE(6,2000)((TECHER(I,J),I=1,3),J=1,TT)
      TT=0
   20 CONTINUE
C
C     IF NO TEACHER SPECIFIED WRITE OUT STAFF AS THE TEACHER NAME
 21   IF(TT.NE.0) WRITE(6,2000)((TECHER(I,J),I=1,3),J=1,TT)
      IF(K.NE.0) GO TO 24
      WRITE(6,2000) (STAFF(I),I=1,3)
   24 L=31
C
C     FIND THE NUMBER OF STUDENTS IN THE REQUEST
      SNUM =RQDATA(29)
      IF(SNUM.LE.0) GO TO 999
      ST=0
      AV12=0
      DO 100 I=1,SNUM
C
C     FIND AND READ IN THE STUDENT NUMBERS SEQUENTIALY
      IF(MOD(30+I,81).NE.0) GO TO 110
      L=L-81
      IF(I.EQ.51)  L=-50
      READ(16'AV16) RQDATA
  110 ST=ST+1
C     SEPARATE THE STUDENT NUMBER FROM THE STUDENT SHORT CODE/
      $NUM= RQDATA(L+I)
      NUM= MOD($NUM,4096)
      IF(NUM.LE.0) GO TO 100
C
C
C     OBTAIN THE STUDENT NAME
      CALL SNAM(NUM)
C
C     LOAD THE STUDENT NAME INTO THE WRITE OUT ARRAY
      DO 120 J=1,3
  120 SNAMES(J,ST)=SAM(J)
C
C     IF THE REQUEST WAS NOT SCHEDULED SKIP THE SHORT CODE PROCESSING
      IF( MSG.EQ.7) GO TO 130
C
C     SEPARATE THE STUDENT NUMBER FROM THE SHORT CODE
      S= $NUM/4096
C
C     INITIALIZE THE SHORT CODE FOR EACH STUDENT
      STIME(ST)=29
      STYME(ST)=29
      IF(S.EQ.0) GO TO 130
      STIME(ST)=30
      STYME(ST)=30
      IF(S.EQ.6) GO TO 130
      STIME(ST)= START
      STYME(ST)= END
      IF(S.EQ.1) STIME(ST)=START+1
      IF(S.EQ.2) STIME(ST)=START+2
      IF(S.EQ.3) STYME(ST)=END-1
      IF(S.EQ.4) STYME(ST)=END-2
      IF(S.EQ.5) STIME(ST)=START+1
      IF(S.EQ.5) STYME(ST)=END-1
C
C     CHECK FOR END OF STUDENTS OR STUDENT BLOCK
  130 IF(ST.LT.SV.AND.I.LT.SNUM) GO TO 100
C
C     WRITE OUT THE STUDENTS IN BLOCKS OF THREE OR FIVE
      IF(I.LE.SV) WRITE(6,2001)
      IF(MSG.EQ.7) WRITE(6,2000)((SNAMES(J,M),J=1,3),M=1,ST)
      IF(MSG.NE.7) WRITE(6,3000)((SNAMES(J,M),J=1,3),(TIME(J,STIME(M))
     1 ,J=1,3),(TYME(J,STYME(M)),J=1,3),M=1,ST)
      ST=0
  100 CONTINUE
```

```
      IF(ST.EQ.0) GO TO 999
      IF(MSG.NE.7) WRITE(6,3000)((SNAMES(J,M),J=1,3),(TIME(J,STIME(M))
     1 ,J=1,3),(TYME(J,STYME(M)),J=1,3),M=1,ST)
      IF(MSG.EQ.7) WRITE(6,2000)((SNAMES(J,M),J=1,3),M=1,ST)
C
C     CHECK FOR END OF THE REQUESTS
  999 IF(B.LT.1250) GO TO 2
      IF(AV17.LT.3) GO TO 1
C
C     FINALIZE
 988  PERIOD=999*32
        WRITE(21) ASEMBL
      ST= 999
      CALL RUMOUT(ST)
      RETURN 1
C***********************************************************************
C***********************************************************************
      END
```

GRPRYT

```
      SUBROUTINE  GRPRYT(*,CRSNO,RQN)
C
C     THIS ROUTINE WRITE OUT ROLLS FOR GROUPS OPEN OPEN-MAY REQUESTA
C     IT ALSO PUTS OUT INFORMATION FOR THE TEACHER SCHEDULES AND ROOM
C     INFORMATION FOR THE  STUDENT SCHEDULES.
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON/BLOK10/NAMESS(141),SAM(3),AV11,AV12,AV13,AV14,AV15,AV16,
     1 AV17,AV21,AV23,AV25
      COMMON /BLOK11/ TYPE(3,7),NOROOM,DATE(2),RQDATA(81),TIME(3,30),
     1 TYME(3,30)
      LOGICAL*1 PER(29)/.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,
     1 .FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,
     2 .FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,
     3 .FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE.,.FALSE./
      REAL*8 NAMESS,SAM
      REAL*8 TYPE,NOROOM,DATE
      REAL*8 NAMES(125),TECHER(3,10),SNAMES(3,5)
      REAL*8 ROOM,COURSE,ROOMS(29),STAFF(3)
      DIMENSION T(29)
      DIMENSION TENM(10),ASEMBL(10),STUNM(2528),STUTIM(2500)
      EQUIVALENCE (NAMES(1),NAMESS(1))
      EQUIVALENCE(ROOM,ASEMBL(3)),(COURSE,ASEMBL(7)),(PERIOD,ASEMBL(1),
     1(PDS,ASEMBL(2))
      EQUIVALENCE (NUMPDS,RQDATA(30)),(ACSTRT,RQDATA(31))
      DATA ZERO/666/
      DATA STAFF/'STAFF   ','        ','        '/
      DATA ROOMS(29)/'NO ROOM '/

C
 1000 FORMAT('2','CLASS ROLL COURSE ',A8,T38,2A8,T61,'TIME   ',6A2,T88,
     1'ROOM - ',A8,T106,3A8,/' ')
 2000 FORMAT(' ',3A8,2X,3A8,2X,3A8,2X,3A8,2X,3A8)
 3000 FORMAT(' ')
 4000 FORMAT(' ',6A2,1X,A8,2X,6A2,1X,A8,2X,6A2,1X,A8,2X,6A2,1X,A8,2X,
     1 6A2,1X,A8)
 5000 FORMAT('2', 'CLASS ROLL COURSE ',A8,T38,2A8,T61,3A8)
C
C
C
      FIND (11'AV11)
C
C     THE TYPE OF REQUEST IS DEDERMINED
      MSG=RQDATA(28)+1
      BETPDS=RQDATA(25)
      ANDPDS=RQDATA(26)
    1 CONTINUE
      AV11= ((CRSNO-1)/125)+1
C
C     PDS IS SET FOR TEACHER INFO WRITE OUT
      PDS=NUMPDS*8+MSG
```

```
C
C     END IS SET FOR  LENGT OF THE GROUPS
      END=NUMPDS-1
C
C     TNUM IS THE NUMBER OF STUDENTS  PLUS THE  ROOM INFO AN THE END OF
C     THE STUDENT  NUMBERS
      TNUM=RQDATA(29)+28
C
C     ********INITIALIZE********
C
C     SNUM IS THE  NUMBER OF STUDENTS IN TH E REQUEST.
      SNUM=RQDATA(29)
      L=31
C
C     **OBTAIN THE  COURSE NAME**
      READ(11'AV11)NAMES
      COURSE= NAMES(CRSNO-(AV11-2)*125)
C
C     OBTAIN THE TEACHER NUMBERS
      DO 10 I=1,10
   10 TENM(I)=RQDATA(10+I)
C
C     READ IN THE STUDENT AND ROOM NUMBERS
      DO 100 I=1,TNUM
      IF(L+I.LT.82) GO TO 140
      L=L-81
      IF(I.LT.132) L=-50
      READ(16'AV16) RQDATA
  140 STDNM=RQDATA(L+I)
      IF(I.GT.SNUM) GO TO 100
      $NUM= STDNM
      STDNM=MOD($NUM,4096)
      IF(MSG.GT.4) GO TO 100
C
C     FIND WHEN THE STUDENT IS SCHEDULED FOR THIS REQUEST
      P=WHEN(STDNM,RQN,BETPDS,ANDPDS)
C
C     SET INDICATOR TO SHOW THAT STUDENT( ) ARE IN THAT  PERIOD
      PER(P)=.TRUE.
C
C     KEEP A RECORD OF WHICH PERID HE WAS SCHEDULED IN
      STUTIM(I)= P
C
C     LOAD THE NUMBERS IN TO AN ARRAY
  100 STUNM(I)= STDNM
C
C     ****ROOM PROSCESSING WHICH ROOM WITH WHICH PERIOD
C     CHECK WERE ANY ROOMS NEED FOR THIS REQUEST
      IF(STUNM(SNUM+1).EQ.0) GO TO 250
      AV12=0
      DO 246 I=1,28
      RM=STUNM(SNUM+I)
      IF(MSG.GT.4.AND.RM.NE.-1) PER(I)=.TRUE.
      IF(RM.EQ.-1.OR..NOT.PER(I)) RM=ZERO
      IF(.NOT.PER(I)) GO TO 245
      IF(RM.EQ.ZERO) GO TO 244
```

```
C
C       READ ROOM NAME
        IF(AV12-2.EQ.(RM-1)/125) GO TO 243
        AV12=((RM-1)/125)+1
        READ(12'AV12) NAMES
  243   ROOMS(I)=NAMES(RM-(AV12-2)*125)
        GO TO 245
C
C       NO ROOM NEEDED
  244   ROOMS(I)=NOROOM
  245   CALL RUMOUT(-RM)
  246   CONTINUE
        IF(MSG.LT.5) GO TO 300
        M=28-END
        DO 800 I=1,M
        IF(.NOT.PER(I)) GO TO 800
        K= I+END
        IF(MSG.EQ.5) GO TO 850
        DO 825 J=I,K
        IF(.NOT.PER(I).OR.I.EQ.M) GO TO 850
        IF(STUNM(SNUM+I).NE.STUNM(SNUM+I+1)) GO TO 850
  825   CONTINUE
        GO TO 800
  850   DO 875 J=I,K
  875   PER(J)=.FALSE.
        PER(I)=.TRUE.
  800   CONTINUE
        K=0
        DO 880 I=1,28
        IF(.NOT.PER(I)) GO TO 880
        K=K+1
        ROOMS(K)= ROOMS(I)
  880   CONTINUE
        GO TO 300
C
C       NO ROOMS NEEDED
  250   DO 255 I=1,28
        CALL RUMOUT(-ZERO)
  255   IF(PER(I)) ROOMS(I)= NOROOM
C
C       ****TEACHER PROCESSING****
  300   AV14=0
        TT=0
C
C       FIND THE TEACHER NUMBERS . ALL POSATIVE.
        DO 350 I=1,10
        IF(TENM(I)) 350,316,309
  309   TT=TT+1
C
C       FIND AND READ IN TEACHER NAME.
        IF(AV14-2.EQ.(TENM(I)-1)/47) GO TO 310
        AV14=((TENM(I)-1)/47)+1
        READ(14'AV14) NAMESS
  310   F= (TENM(I)-(AV14-2)*47)*3-3
        DO 315 H=1,3
```

```
 315  TECHER(H,TT)= NAMESS(F+H)
      GO TO 317
 316  IF(TT.NE.0) GO TO 399
C
C     NO TEACHER NAME  LOAD IN STAFF
      TT=1
      DO 352 K= 1,3
 352  TECHER(K,TT)= STAFF(K)
C
C     WRITE OUT OF TEACHER INFORMATION
  317 DO 320 J=1,28
      IF(.NOT.PER(J)) GO TO 320
      PERIOD=TENM(I)*32+J
      ROOM= ROOMS(J)
      WRITE(21) ASEMBL
  320 CONTINUE
      IF(TENM(I).EQ.0) GO TO 399
  350 CONTINUE
  399 L=1
C     CHECK FOR MAY AND OPEN MAY
      IF(MSG.GT.4) GO TO 600
  400 DO 405 P=L,29
      IF(PER(P)) GO TO 410
  405 CONTINUE
      GO TO 610
  410 L=P+1
      PER(P)=.FALSE.
C     CHECK FOR MAY AND OPEN MAY
      IF(MSG.GT.4) GO TO 605
      IF(P.EQ.29) MSG=7
      IF(MSG.EQ.7) END=0
C
C     WRITE OUT HEADING AND TEACHER NAMES FOR GROUP,OR OPEN REQUESTS
      WRITE(6,1000) COURSE,DATE(1),DATE(2),(TIME(I,P),I=1,3),(TYPE(I,P+
     1 END),I=1,3), ROOMS(P),(TYPE(I,MSG),I=1,3)
      WRITE(6,2000)((TECHER(I,J),I=1,3),J=1,TT)
      WRITE(6,3000)
  500 AV13=0
      K=0
      DO 590 I=1,SNUM
C
C     CHECK ROR MAY OR OPEN MAY
      IF(MSG.EQ.7) GO TO 508
      IF(MSG.GT.4) GO TO 509
 508  IF(STUTIM(I).NE.P) GO TO 590
C
C     OBTAIN STUDENT NAME AND LOAD IN WRITE OUT ARRAY
  509 CALL SNAM(STUNM(I))
      K=K+1
      DO 510 J=1,3
 510  SNAMES(J,K)= SAM(J)
C
```

```
C     WRITE OUT STUDENT NA ES IN GOUPS OF FIVE
      IF(K.LT.5.AND.I.LT.SNUM) GO TO 590
      WRITE(6,2000)((SNAMES(Z,J),Z=1,3),J=1,K)
      K=0
  590 CONTINUE
      IF(K.NE.0) WRITE(6,2000)((SNAMES(Z,J),Z=1,3),J=1,K)
C     CHECK FOR END OF WORK
      IF(MSG.GT.4) RETURN 1
      GO TO 400
C
C     WRITE OUT HEADING AND TEACHER NAMES FOR MAY AND OPEN MAY
 600  WRITE(6,5000) COURSE,DATE,(TYPE(I,MSG),I=1,3)
      WRITE(6,2000)((TECHER(I,J),I=1,3),J=1,TT)
      M=0
      GO TO 400
C
C     OBTAIN TIMES OF THE ROOMS FOR MAY AND OPEN MAY
  605 M=M+1
      T(M)=P
      GO TO 400
C
C     CHECK FOR END OF WORK
  610 IF(MSG.LT.5) RETURN 1
C
C***************************************************************************
      IF(M.EQ.0) GO TO 500
C***************************************************************************
C     WRITE OUT THE ROOMS AND TIMES FOR MAY AND OPEN MAY
      WRITE(6,4000)((TIME(I,T(K)),I=1,3),(TYME(I,T(K)+END),I=1,3),
     1 ROOMS(K),K=1,M)
      WRITE(6,3000)
      GO TO 500
C***************************************************************************
C***************************************************************************
      END
```

WHEN

```
      FUNCTION WHEN(STDNM,REQNUM,BETPDS,ANDPDS)
C      THIS  FUNCTION IS USED BY GRPRYT TO FIND WHICH PERIOD OF THE
C        DAY A STUDENT HAS BEEN SCHEDULED IN.
      IMPLICIT INTEGER*2(A-Z), INTEGER*4($)
      COMMON /BLOK05/ PART, STDINF(28,256), STDLST(2500), INSEC, IN, UP
      LOGICAL*1 IN,UP
      INTEGER*4 POSNUM, PART
 10   POSNUM=STDLST(STDNM)
      IF(POSNUM.LE.0) GO TO 40
      REQSEC = ((POSNUM-1)/PART) + 1
      IF(REQSEC.EQ.INSEC) GO TO 20
      INSEC= REQSEC
      READ(25'(REQSEC*4-3)) STDINF
 20   K= POSNUM-(INSEC-1)*PART
      DO 30 WHEN=BETPDS,ANDPDS
      IF(STDINF(WHEN,K).EQ.REQNUM) RETURN
   30 CONTINUE
 40   WHEN=29
      RETURN
      END
```

SNAM

```
      SUBROUTINE SNAM(NUM)
C     THIS ROUTINE IS USED TO RETRIEVE STUDENT NAMES FROM FILE.
C     'NUM' CONTAINS THE STUDENT NUMBER
C    'SAM' WILL CONTAIN THE STUDENTS NAME
C     BEFORE INITIAL USE AV13 MUST BE 1 OR 0
      IMPLICIT INTEGER*2 (A-Z) ,INTEGER*4($)
      REAL*8 NAMESS,SAM
      COMMON /BLOK10/NAMESS(141),SAM(3),AV11,AV12,AV13,AV14,AV15,AV16,
     1 AV17,AV21,AV23,AV25
C
C
C     CHECK TO SEE IF IT IS ALREADY IN CORE
      IF(AV13.EQ.((NUM-1)/47)+2) GO TO 10
      AV13=((NUM-1)/47)+1
      READ(13'AV13) NAMESS
 10   K=(NUM-(AV13-2)*47)*3-3
      DO 20 I=1,3
 20   SAM(I)=NAMESS(K+I)
      RETURN
      END
```

RUMOUT

```
      SUBROUTINE RUMOUT(RMN)
C
      IMPLICIT INTEGER*2 (A-Z) ,INTEGER*4($)
      COMMON /BLOK13/ RUMINF(1250),MAYINF(28,11),IN,IN2,CRSE(2500)
C
C
      DATA CNT/1/,P/1/,K/1/,AV17/5/,AV23/1/,MCNT/-1/
C
      IF(RMN.EQ.999) GO TO 25
      IF(RMN.LT.0) GO TO 20
      RUMINF(CNT)=RMN
      CNT=CNT+1
 15   IF(CNT.LT.1251) RETURN
      WRITE(17'AV17) RUMINF
      CNT=1
      AV17=AV17+1
      RETURN
 20   MAYINF(K,P)=-RMN
      IF(RMN.EQ.-666) MAYINF(K,P)=0
      K=K+1
      IF(K.LT.29) RETURN
      RUMONF(CNT)=MCNT
      MCNT=MCNT-1
      K=1
      P=P+1
      CNT= CNT+1
      IF(P.LE.11) GO TO 15
      WRITE(23'AV23)MAYINF
      AV23= AV23+1
      P=1
      SUBROUTINE  STURYT
C         THIS ROUTINE WRITES OUT THE STUDENT ROLLS.
C         CLASS CONTAINS A RECORD OF THE FIRST AND LAST STUDENT NUMBER
C         IN EACH GRADE (CLASS).
C
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON /BLOK10/NAMESS(141),SAM(3),AV11,AV12,AV13,AV14,AV15,AV16,
     1 AV17,AV21,AV23,AV25
      COMMON /BLOK11/ TYPE(3,7),NOROOM,DATE(2),RQDATA(81),TIME(3,30),
     1 TYME(3,30)
      COMMON/BLOK12/ CRSS(2,28), RMSS(2,28),STUNUM
      REAL*8 NAMESS,SAM
      REAL*8 TYPE,NOROOM,DATE
      REAL*8 MESSAG(20),NONAME,RUMS(28),KORS(28),NAMES(125),BLANK
      DIMENSION MASLOK(1025),CLASS(25),DISMBL(4),RMN(28),CRN(28)
      DIMENSION START(28),PDS(28)
      EQUIVALENCE (MASLOK(1002),CLASS(1)),(DISMBL(1),MASLOK(998)    )
      EQUIVALENCE (RMN(1),MASLOK( 970)),(CRN(1),MASLOK(942))
      EQUIVALENCE (START(1),MASLOK(914)),(PDS(28),MASLOK(886))
      EQUIVALENCE (BLANK,TYPE(1,1)),(CRNO,DISMBL(3)),(ROOM,DISMBL(4))
      EQUIVALENCE (NUM,DISMBL(2)),(SN,DISMBL(1))
      DATA NONAME /'NO NAME '/
C
      LOGICAL*1 HEAD
C
      NAMELIST/OUTNAM/ I,AV12,AV11,CRSS,RMSS
 1000 FORMAT(' ***** YOU HAVE NOT BEEN ASSIGNED TO A GRADE *****')
 2000 FORMAT(10A8)
 3000 FORMAT('2',3A8,T43,'STUDENT SCHEDULE  ',2A8,T122,'GRADE ',I3,//,
     1T1,4(5X,'TIME     COURSE      ROOM',5X),/,' ')
 4000 FORMAT(' ',4(I3,1X,3A2,2X,A8,2X,A8,3X))
 5000 FORMAT('0',T42,'YOU MAY CHOOSE FROM THE FOLLOWING CLASSES',/,' ')
 6000 FORMAT(' ',T26,10A8,/,' ',T26,10A8)
C
C
    1 CONTINUE
      READ(15'1) MASLOK
      CLASS(25)= 0
      CLA=24
      STUNUM= 1
C
C         READ IN FIRST MAY DATA RECORD. (THESE RECORDS ARE 8 BYTES LONG)
      READ(20,END=889) DISMBL
    2 AV13= 0
C
C         OBTAIN STUDENT NAME AND CHECK FOR NO NAME (IGNORE THESE)
    5 CALL SNAM(STUNUM)
      IF(SAM(1).EQ.NONAME) GO TO 999
C
C         CHECK TO SEE IF STUDENT IS IN THE CLASS NOW BEING WORKED ON.
      IF(STUNUM.GE.CLASS(CLA).AND.STUNUM.LE.CLASS(CLA+1)) GO TO 20
```

```
C
C         FIND THE CORRECT CLASS.
      DO 10 I=1,24,2
      IF(STUNUM.GE.CLASS(I).AND.STUNUM.LE.CLASS(I+1)) GO TO 15
   10 CONTINUE
C
C         ERROR--STUDENT IS NOT IN A CLASS.
      CLA=0
      GO TO  20
   15 CLA= I
C
C         READ IN A MESSAGE FOR THE CLASS WHICH IS TO BE WORKED ON.
      READ(8,2000) MESSAG
C
C         OBTAIN A STUDENT'S SCHEDULE AND ROOM INFORMATION FOR EACH
C         COURSE HE IS SCHEDULED IN. THIS INFORMATION RETURNS IN RMSS
C         AND CRSS.
   20 CALL  WHICH
C
C         STATEMENTS 20+2 THRU 30 OBTAIN AND STORE COURSE NAMES.
C         IF THE STUDENT IS NOT SCHEDULED FOR A COURSE DURING A PERIOD, A
C         BLANK IS STORED IN PLACE OF A COURSE NAME.
      AV11= 0
      DO 30 I=1,28
      IF(CRSS(1,I).EQ.0) GO TO 29
      IF(AV11-2.EQ.(CRSS(1,I)-1)/125) GO TO 25
      AV11=((CRSS(1,I)-1)/125)+1
      READ(11'AV11) NAMES
   25 KORS(CRSS(2,I))= NAMES(CRSS(1,I)-(AV11-2)*125)
      IF((CRSS(1,I)-(AV11-2)*125).GT.125) WRITE(3,CUTNAM)
      GO TO 30
   29 KORS(CRSS(2,I))= BLANK
   30 CONTINUE
C
C         STORE ROOM NAMES.
C         IF A COURSE NAME IS NOT RECORDED FOR A PERIOD, A BLANK IS
C         STORED FOR A ROOM NAME. IF A COURSE NAME IS RECORDED FOR A PERIOD
C         AND THE ROOM NUMBER IS ZERO FOR THAT PERIOD, 'NO ROOM' IS
C         STORED AS THE ROOM NAME.
      AV12=0
      DO 40 I=1,28
      IF(RMSS(1,I).EQ.0) GO TO 39
      IF(AV12-2.EQ.(RMSS(1,I)-1)/125) GO TO 35
      AV12=((RMSS(1,I)-1)/125)+1
      READ(12'AV12) NAMES
   35 RUMS(RMSS(2,I))= NAMES(RMSS(1,I)-(AV12-2)*125)
      IF((RMSS(1,I)-(AV12-2)*125).GT.125) WRITE(3,CUTNAM)
      GO TO 40
   39 RUMS(RMSS(2,I))= BLANK
      IF(KORS(RMSS(2,I)).NE.BLANK) RUMS(RMSS(2,I))= NOROOM
   40 CONTINUE
      LIM=(CLA+1)/2
C
C         WRITE OUT THE HEADING.
      WRITE(6,3000)(SAM(I),I=1,3),DATE(1),DATE(2), LIM
```

```
C         WRITE OUT THE STUDENT'S SCHEDULE.
      DO 45 J=1,7
   45 WRITE(6,4000)(M,(TIME(L,M),L=1,3),KORS(M),RUMS(M),M=J,28,7)
      LIM=0
      IF(CLA.EQ.0) WRITE(6,1000)
      HEAD=.TRUE.
C
C         CHECK FOR MAY DATA.
   50 IF(SN.NE.STUNUM) IF(LIM)90,988,90
C
C         OBTAIN START PERIOD OF A MAY COURSE FOR THE STUDENT.
      STRT= NUM/32
C
C         CHECK TO SEE IF START PERIOD IS OPEN IN THE STUDENT'S SCHEDULE.
      IF(RUMS(STRT).NE.BLANK) GO TO 80
C
C         OBTAIN THE NUMBER OF PERIODS THE MAY COURSE EXTENDS.
      $NUM= NUM
      NUM= MOD($NUM,32) + STRT-1
C
C         CHECK--ARE ALL PERIODS FOR MAY COURSE OPEN ON STUDENT'S
C         SCHEDULE.
      DO 55 I=STRT,NUM
      IF(RUMS(I).NE.BLANK) GO TO 80
   55 CONTINUE
C
C         COUNT THE NUMBER OF MAY COURSES THE STUDENT CAN USE.
      LIM=LIM+1
C
C         STORE MAY INFORMATION.
      CRN(LIM)= CRNO
      RMN(LIM)= ROOM
      START(LIM)=STRT
      PDS(LIM)= NUM-STRT
C
      IF(LIM.EQ.28) GO TO 90
C         READ IN A MAY DATA RECORD.
   80 READ(20,END=888) DISMBL
      GO TO 50
C
C         SET UP A DELIMITER. THE END OF THE MAY DATA HAS BEEN
C         ENCOUNTERED.
  888 SN=0
      GO TO 50
C
C         OBTAIN AND STORE MAY COURSE ROOM NAMES.
   90 NMCHK=0
      DO 95 I=1,LIM
      IF(RMN(I).EQ.0) GO TO 93
      IF(RMN(I).EQ.NMCHK) GO TO 92
      IF(AV12-2.EQ.(RMN(I)-1)/125) GO TO 91
      AV12=((RMN(I)-1)/125)+1
      READ(12'AV12) NAMES
   91 RUMS(I)= NAMES(RMN(I)-(AV12-2)*125)
```

```
      GO TO 95
   92 RUMS(I)=RUMS(I-1)
      GO TO 95
   93 RUMS(I)=NOROOM
   95 NMCHK= RMN(I)
      NMCHK=0
C
C        OBTAIN AND STORE MAY COURSE COURSE NAMES.
      AV11=0
      DO 100 I=1,LIM
      IF(CRN(I).EQ.NMCHK) GO TO 108
      IF(AV11-2.EQ.(CRN(I)-1)/125) GO TO 106
      AV11=((CRN(I)-1)/125)+1
      READ(11'AV11) NAMES
  106 KORS(I)= NAMES(CRN(I)-(AV11-2)*125)
      GO TO 100
  108 KORS(I)=KORS(I-1)
  100 NMCHK= CRN(I)
C
C        WRITE MAY HEADING AND MAY SCHEDULE.
      IF(HEAD) WRITE(6,5000)
      IF(HEAD) HEAD=.FALSE.
      WRITE(6,4000)(PDS(I),(TIME(L,START(I)),L=1,3),KORS(I),RUMS(I),
     1 I=1,LIM)
      IF(LIM.EQ.28) GO TO 80
C
C        WRITE CLASS MESSAGE.
 988  IF(CLA.NE.0) WRITE(6,6000) MESSAG
      IF(CLA.EQ.0) CLA=24
C
C        INCREMENT AND CHECK STUDENT NUMBER.
  999 STUNUM=STUNUM+1
      IF(STUNUM.LE.2500) GO TO 5
      RETURN
C
C        SET UP A DELIMITER, THE END OF THE MAY DATA HAS BEEN
C        ENCOUNTERED. (THERE WAS NO MAY DATA)
  889 SN=0
      GO TO  2
      END
```

WHATRM

```
      FUNCTION WHATRM(RQN,I)
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON /BLOK13/ RUMINF(1250), MAYINF(28,11),IN,IN2,CRSE(2500)
      NAMELIST/NAMEIT/XRQN,XI,POINT,IN,A,K,WHATRM
      NAMELIST/ANAME/ XRQN,XI,POINT,IN,A,K,IN2,WHATRN
100   FORMAT(' SUBROUTINE WHATRM'/(1X,14I8))
      XRQN=RQN
      XI=I
C
C
    1 CONTINUE
      POINT= ((RQN-1)/1250)+5
      IF(POINT.EQ.IN) GO TO 20
      READ(17'POINT) RUMINF
      IN= POINT
   20 K= RQN-(POINT-5)*1250
      A=RUMINF(K)
      IF(A.LT.0) GO TO 30
      WHATRM= A
      IF(WHATRM.LE.500.AND.WHATRM.GE.0) RETURN
      WRITE(3,NAMEIT)
      WRITE(3,100)RUMINF
      WHATRM=0
      RETURN
 30   A=-A
      POINT=((A-1)/ 11)+1
      IF(POINT.EQ.IN2) GO TO 40
      READ(23'POINT) MAYINF
      IN2=POINT
 40   K=A-(POINT-1)*11
      WHATRM= MAYINF(I,K)
      IF(WHATRM.LE.500.AND.WHATRM.GE.0) RETURN
      WRITE(3,ANAME)
      WRITE(3,100)MAYINF
      WHATRM=0
      RETURN
      END
```

WHICH

```
      SUBROUTINE WHICH
      IMPLICIT  INTEGER*2(A-Z),INTEGER*4($)
      COMMON /BLOK13/ RUMINF(1250), MAYINF(28,11),IN,IN2,CRSE(2500)
      COMMON/BLOK12/ CRSS(2,28), RMSS(2,28),STUNUM
      COMMON /BLOK05/ PART, STDINF(28,256), STDLST(2500), INSEC
      INTEGER*4 CORS(28),ROOMS(28),BIGSRT
      INTEGER*4 POSNUM,PART
      EQUIVALENCE(CRSS(1,1),CORS(1)),(RMSS(1,1),ROOMS(1))
C
C
      POSNUM= STDLST(STUNUM)
      REQSEC= ((POSNUM-1)/PART)+1
      IF(REQSEC.EQ.INSEC) GO TO 20
      INSEC= REQSEC
      READ(25'REQSEC) STDINF
   20 K= POSNUM-(INSEC-1)*PART
      DO 25 I=1,28
      RQN=STDINF(I,K)
      IF (RQN.EQ.0) GO TO 23
      CRSS(1,I)=CRSE(RQN)
      RMSS(1,I)=WHATRM(RQN,I)
      GO TO 24
   23 CRSS(1,I)=0
      RMSS(1,I)=0
   24 CRSS(2,I)=I
   25 RMSS(2,I)=I
      CALL BIGSRT(CORS)
      CALL BIGSRT(ROOMS)
      RETURN
      END
      SUBROUTINE BIGSRT(A)
C
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      INTEGER*4 A(28),TEMP
      DO 10 J=1,27
      I=J
    9 IF (A(I).LE.A(I+1)) GO TO 10
      TEMP=A(I)
      A(I)=A(I+1)
      A(I+1)=TEMP
      I=I-1
      IF (I) 10,10,9
   10 CONTINUE
      RETURN
      END

      BLOCK DATA
C     BLOCK DATA 11
C         BLOCK DATA 11 IS USED TO LOAD INTO COMMON PERIOD START TIMES,
C         END TIMES, AND INFORMATION MESSAGES AS TO REQUEST TYPE.
C         THE INFORMATION IN BLOK11 IS USED BY ALL SCHEDULE WRITEOUT
C         ROUTINES.
C
C
      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)
      COMMON /BLOK11/ TYPE(3,7),NOROOM,DATE(2),RQDATA(81),TIME(3,30),
     1 TYME(3,30)
      REAL*8 TYPE,NOROOM,DATE
C
C
      DATA  TIME /' 8 30 ',' 8 45 ',' 9 00 ',' 9 15 ',' 9 30 ',' 9 45 ',
     1'10 00 ','10 15 ','10 30 ','10 45 ','11 00 ','11 15 ','11 30',
     2'11 45 ','12 00 ','12 15 ','12 30 ','12 45 ',' 1 00 ',' 1 15 ',
     3' 1 30 ',' 1 45 ',' 2 00 ',' 2 15 ',' 2 30 ',' 2 45 ',' 3 00 ',
     4' 3 15 ','      ','UNAVAI'/
C
C
      DATA  TYME /'- 8 40','- 8 55','- 9 10','- 9 25','- 9 40','-9 55',
     1'-10 10','-10 25','-10 40','-10 55','-11 10','-11 25','-11 40',
     2'-11 55','-12 10','-12 25','-12 40','-12 55','- 1 10','- 1 25',
     3'- 1 40','-1 55','- 2 10','- 2 25','- 2 40','- 2 55','- 3 10',
     4'- 3 25','      ','LABLE '/
C
C
      DATA TYPE  /'                        ','   NO-PRIORITY REQUEST  ',
     1'      GROUP REQUEST     ','      OPEN REQUEST      ',
     2'       MAY REQUEST      ','    OPEN-MAY REQUEST    ',
     3'*COULD NOT BE SCHEDULED*'/
C
      DATA NOROOM /'NO ROOM '/
      END
```

Job Step 12

TECRYT

```
~~      IMPLICIT INTEGER*2 (A-Z),INTEGER*4($)~~
      COMMON /BLOK11/ TYPE(3,7),NOROOM,DATE(2),RQDATA(81),TIME(3,30),
     1 TYME(3,30)
~~      LOGICAL*1 CHECK~~
      REAL*8 NAMESS
      REAL*8 TYPE,NOROOM,DATE
~~      REAL*8 CRSARY(28),RMARY(28),MESSAG(20),ROOM,COURSE~~
      REAL*8 DITTO
      REAL*8 STAFF(3)
~~      DIMENSION NAMESS(141)~~
      DIMENSION MSG(28),DISMBL(10)
      EQUIVALENCE (DISMBL(3),ROOM),(DISMBL(7),COURSE)
~~      DATA DITTO /8H     ''   /~~
      DATA STAFF/'STAFF   ','        ','        '/
      DATA CHECK/.FALSE./
~~C~~
C
 1000 FORMAT ('2',3A8,2X,2A8,3X,'TECHER/ADMINISTRATOR SCHEDULE'/,'0'
~~     1  T6, 'TIME  COURSE',3X,'ROOM',2(25X, 'TIME  COURSE',3X,'ROOM'),/,~~
     2 '0')
 1001 FORMAT (2A8)
~~ 2000 FORMAT( 10A8/10A8)~~
 3000 FORMAT(' ',3(I2,1X,3A2,1X,A8,1X,A8,1X,3A8))
 4000 FORMAT(T26,10A8,/,T26,10A8)
~~C~~
      NAMELIST/AARG/ CHECK
C
C
      DEFINE FILE 14(11,282,U,AV14)
C
      READ(8,1001) DATE
      AV14=0
      READ(8,2000) MESSAG
~~      ASSIGN 5 TO $J~~
    1 DO 4 I=1,28
      MSG(I)=1
~~      CRSARY(I)= TYPE(1,1)~~
    4 RMARY(I)= TYPE(1,1)
      ASSIGN 10 TO $K
~~      GO TO $J,(5,10)~~
C5    READ(21) DISMBL
5     READ(21,END=99)DISMBL
~~  994 CONTINUE~~
      TN= DISMBL(1)/32
  992 CONTINUE
~~      IF(TN.EQ.999) GO TO  999~~
      GO TO $K,(10,15)
   10 TEMP=TN
~~      ASSIGN 15 TO $K~~
      GO TO 16
   15 IF(TN.NE.TEMP) GO TO 100
   16 $DIS= DISMBL(1)
      PD= MOD($DIS,32)
      DIT= (DISMBL(2)/8)-1
  991 CONTINUE
      $DIS=DISMBL(2)
      MSG(PD)= MOD($DIS,8)
      CRSARY(PD)= COURSE
      RMARY(PD) = ROOM
      IF(DIT.LE.0) GO TO 5
      DO 20 I=1,DIT
      CRSARY(PD+I)= DITTO
   20 RMARY (PD+I)= DITTO
      GO TO 5
  100 IF(TN.EQ.0) GO TO 195
      IF(AV14-2.EQ.(TEMP-1)/47) GO TO 105
      AV14= ((TEMP-1)/47)+1
      READ(14'AV14) NAMESS
  105 K= (TEMP-(AV14-2)*47)*3-3
      WRITE(6,1000)(NAMESS(K+I),I=1,3),DATE(1),DATE(2)
  195 IF(TEMP.EQ.0)WRITE(6,1000)(STAFF(I),I=1,3),DATE(1),DATE(2)
      LIM=1
      DO 200 J=1,10
      WRITE(6,3000)(M,(TIME(L,M),L=1,3),CRSARY(M),RMARY(M),(TYPE(L,MSG
     1 (M)),L=1,3),M= LIM,28,10)
  200 LIM=LIM+1
      WRITE(6,4000) MESSAG
  993 CONTINUE
C     IF(CHECK) CALL EXIT
      IF(CHECK) WRITE(6,AARG)
      ASSIGN 10 TO $J
      GO TO 1
  999 CHECK= .TRUE.
      GO TO 100
99    STOP
      END
```

Job Step 13

STATS

```
C   STATISTICS ROUTINE -
       IMPLICIT INTEGER*4  (A-Z)
      INTEGER*2 FILE17,CRSE,SLIST,INFO,OUT,TCHR,NUMPDS,TYPE,NUMSTU,
     C TNUM,TCNUM,OUTPUT
     C RLONG,ACSTRT,CNUM,SNUM,TRQST,RQSC,RQSP,RST,PST,PRQ,TPRD,ACC,TGRP
      DIMENSION FILE17(5000), CRSE(2500), SLIST(2500), INFO(31), OUT(10)
     1, TCHR(10),OUTPUT(19)
      INTEGER*4   RQSTCT(6), STUDRQ(6), STPDRQ(6), BUMPED(6), SRBUMP(6),
     2 SPBUMP(6), RQSCH(6), SRSCH(6), SPSCH(6), SHORTS(6), PSHORT(6)
      EQUIVALENCE (FILE17(2501),CRSE), (INFO(11),TCHR), (INFO(12),NUMPDS
     1), (INFO(28),TYPE), (INFO(29),NUMSTU), (INFO(30),RLONG),
     2(INFO(31),ACSTRT), (OUT,CNUM), (OUT(2),SNUM), (OUT(3),TRQST),
     3(OUT(4),RQSC), (OUT(5),RQSP), (OUT(6),RST), (OUT(7),PST),
     4(OUT(8),PRQ), (OUT(9),TPRD), (OUT(10),ACC), (INFO(27),TGRP)
     5,(TNUM,OUT),(TCNUM,OUT(2))
      LOGICAL  SKED, NOPRIO,MAY
      DATA RQSTCT, STUDRQ, STPDRQ, BUMPED, SRBUMP, SPBUMP, RQSCH, SRSCH,
     1SHORTS, PSHORT/60* 0/, ACC/1/
 1001 FORMAT(1H120X'REGULAR  REQUESTS')
 1002 FORMAT(1H020X'OPEN  REQUESTS')
 1003 FORMAT(1H020X'GROUP  REQUESTS')
 1004 FORMAT(1H020X'OPEN-MAY  REQUESTS')
 1005 FORMAT(1H020X'GROUP-MAY  REQUESTS')
 1006 FORMAT(1H020X'NO-PRIORITY  REQUESTS')
 1010 FORMAT(10X'NUMBER REQUESTED'19X,I10)
 1011 FORMAT(10X'NUMBER REQUESTS SCHEDULED'10X,I10)
 1012 FORMAT(10X'NUMBER REQUESTS SHORTENED'10X,I10)
 1014 FORMAT(10X'NUMBER PERIODS SHORTENED'11X,I10)
 1016 FORMAT(10X'NUMBER REQUESTS BUMPED'13X,I10)
 1018 FORMAT(10X'NUMBER OF STUDENT-PERIODS REQUESTED'I10)
 1020 FORMAT(10X'NUMBER OF STUDENT-PERIODS BUMPED',3X,I10)
 1025 FORMAT(1H1,14X'NUMBER OF NORMAL REQUESTS MADE'10X,I10)
 1026 FORMAT(15X'NUMBER OF NORMAL REQUESTS SCHEDULED',5X,I10)
 1027 FORMAT(15X'NUMBER OF NORMAL REQUESTS BUMPED',8X,I10)
 1028 FORMAT(15X'NUMBER OF NO-PRIORITY REQUESTS SCHEDULED',I10)
 1030 FORMAT(1H0,4X,'* THE FOLLOWING TOTALS DO NOT INCLUDE MAY OR NO-PRI
     $ORITY TYPE REQUEST, EXCEPT WHERE INDICATED')
 1032 FORMAT(1H0,9X'TOTAL NUMBER OF STUDENT-PERIODS REQUESTED',13X,I10)
 1034 FORMAT(10X'TOTAL NUMBER OF STUDENT-PERIODS RECEIVED',14X,I10)
 1036 FORMAT(10X'TOTAL NUMBER OF STUDENT-PERIODS BUMPED',16X,I10)
 1038 FORMAT(10X'TOTAL NUMBER OF STUDENT-PERIODS FILLED BY NO-PRIORITY'
     $1X,I10)
 1040 FORMAT(1H0,9X'TOTAL NUMBER OF STUDENT-REQUESTS MADE',17X,I10)
 1042 FORMAT(10X'TOTAL NUMBER OF STUDENT-REQUESTS RECEIVED'13X,I10)
 1044 FORMAT(10X'TOTAL NUMBER OF STUDENT-REQUESTS BUMPED',15X,I10)
 1046 FORMAT(10X'TOTAL NUMBER OF STUDENT-REQUESTS FILLED BY NO-PRIORITY'
     $,I10)
 1050 FORMAT(1H0,9X'PERCENT OF REQUESTED STUDENT-PERIODS RECEIVED',I10)
 1052 FORMAT(10X'PERCENT OF REQUESTED STUDENT-PERIODS BUMPED'2X,I10)
      DEFINE FILE 16(600,162,L,AV16)
      DEFINE FILE 17 (8,625,U,AV17)
      AV17= 1
      READ(17'AV17)FILE17
      DO 300 I=1,1250
      IF (FILE17(I).EQ.0) GO TO 350
      AV16= FILE17(I)
      SKED= .FALSE.
      READ(16'AV16) INFO,(SLIST(I),I=1,NUMSTU)
      IF (ACSTRT.GT.0) SKED= .TRUE.
      TYPE= TYPE+ 1
      ROSTCT(TYPE)= ROSTCT(TYPE)+ 1
      STUDRQ(TYPE)= STUDRQ(TYPE)+ NUMSTU
      STPDRQ(TYPE)= STPDRQ(TYPE)+ NUMSTU* NUMPDS
      CNUM= CRSE(I)
      IF (.NOT.SKED) GO TO 250
  200 IF (TYPE.GE.5) GO TO 300
      SHORT= NUMPDS- RLONG
      SPBUMP(TYPE)= SPBUMP(TYPE)+ NUMSTU* SHORT
      TRQST= 1
      PRQ= NUMPDS
      MAY= TYPE.EQ.3.OR.TYPE.EQ.5
      DO 211 J=14,19
  211 OUTPUT(J)=0
      OUTPUT(13)=RLONG
      IF(.NOT.SKED) OUTPUT(19)=1
      IF(SKED.AND.MAY) OUTPUT(15)=RLONG
      DO  245 J=1,NUMSTU
      STNUM= SLIST(J)
      CODE= STNUM/ 4096+ 1
      STNUM= STNUM-((CODE-1)* 4096)
      OUTPUT(1)=STNUM
      IF(MAY.OR .NOT.SKED) GO TO 205
      OUTPUT(14)=RLONG
      OUTPUT(17)=1
      IF(CODE.EQ.1) GO TO 207
  206 OUTPUT(14)=0
      OUTPUT(19)=1
      OUTPUT(17)=0
      OUTPUT(18)=0
  205 IF(CODE.EQ.7) GO TO 207
      OUTPUT(14)=RLONG-1
      OUTPUT(19)=0
      OUTPUT(18)=1
      IF(CODE.EQ.1.OR.CODE.EQ.3) GO TO 207
      OUTPUT(14)=RLONG-2
  207 WRITE(40) OUTPUT
      RSHORT=0
      RQSC= 1
      RQBP= 0
      IF (SHORT.GT.0) RSHORT= 1
      SHORTS(TYPE)= SHORTS(TYPE)+ RSHORT
      PSHORT(TYPE)=PSHORT(TYPE)+SHORT
      GO TO  (210,220,230,220,230,230,240),CODE
  210 TPRD= RLONG
      RST= RSHORT
      PST= SHORT
      GO TO 245
  220 TPRD= RLONG- 1
      RST= 1+ RSHORT
      PST= 1+ SHORT
      SPBUMP(TYPE)= SPBUMP(TYPE)+ 1
      GO TO 245
  230 TPRD= RLONG- 2
      RST= 1+ RSHORT
      PST= 2+ SHORT
      SPBUMP(TYPE)= SPBUMP(TYPE)+ 2
      GO TO 245
  240 TPRD= 0
      RST= 0
      PST= 0
      SPBUMP(TYPE)= SPBUMP(TYPE)+ NUMPDS
      SRBUMP(TYPE)= SRBUMP(TYPE)+ 1
      RQBP= 1
      RQSC= 0
  245 WRITE (29)OUT
      GO TO 260
  250 BUMPED(TYPE)= BUMPED(TYPE)+ 1
      SRBUMP (TYPE)= BUMPED(TYPE)+ NUMSTU
      SPBUMP(TYPE)= SPBUMP(TYPE)+ NUMSTU* NUMPDS
      IF (TYPE.GE.5)  GO TO 260
      TRQST= 1
      IF(TYPE.EQ.2)TRQST= 0
      RQSC= 0
      RQBP= 1
      RST= 0
      PST= 0
      PRQ=  NUMPDS
      TPRD= 0
      DO 255 J=1,NUMSTU
      STNUM= SLIST(J)- (SLIST(J)/4096)* 4096
  255 WRITE(29) OUT
      GO TO 260
  260 IF (SKED) GO TO 270
      TRQST= 1
      IF (TYPE.EQ.2) TRQST= 0
      RQSC= 0
      RQBP= 1
      RST= 0
      PST= 0
      PRQ= NUMPDS
      IF(TYPE.EQ.3.OR.TYPE.EQ.4) PRQ=TGRP
      TPRD= 0
      GO TO 280
  270 TRQST= 1
      RQSC= 1
      RQBP= 0
      RST=0
      IF (SHORT.GT.0) RST=1
      PST= SHORT
      PRQ = NUMPDS
      IF(TYPE.EQ.3.OR.TYPE.EQ.4) PRQ=TGRP
      TPRD= 0
  280 TCNUM= CRSE(I)
      DO 285 J=1,10
      IF(TCHR(J).EQ.0) GO TO 300
      TNUM= TCHR(J)+ 1000
  285 WRITE (29) OUT
  300 CONTINUE
  350 DO 360 I= 1,6
      RQSCH(I)= RQSTCT(I)- BUMPED(I)
      SRSCH(I)= STUDRQ(I)- SRBUMP(I)
      SPSCH(I)= STPDRQ(I)- SPBUMP(I)
  360 CONTINUE
      CNUM= 9999
      SNUM= 9999
      WRITE(29) OUT
      WRITE (6,1001)
      WRITE (6,1010) RQSTCT(1)
      WRITE (6,1011) RQSCH(1)
      WRITE (6,1012) SHORTS(1)
      WRITE (6,1014) PSHORT(1)
      WRITE (6,1016) BUMPED(1)
      WRITE (6,1018) STPDRQ(1)
      WRITE (6,1020) SPBUMP(1)
      WRITE (6,1002)
      WRITE (6,1010) RQSTCT(4)
      WRITE (6,1011) RQSCH(4)
      WRITE (6,1016) BUMPED(4)
      WRITE (6,1018) STPDRQ(4)
      WRITE (6,1020) SPBUMP(4)
      WRITE (6,1003)
      WRITE (6,1010) RQSTCT(3)
      WRITE (6,1011) RQSCH(3)
      WRITE (6,1016) BUMPED(3)
      WRITE (6,1018) STPDRQ(3)
      WRITE (6,1020) SPBUMP(3)
      WRITE (6,1004)
      WRITE (6,1010) RQSTCT(6)
      WRITE (6,1018) STPDRQ(6)
      WRITE (6,1005)
      WRITE (6,1010) RQSTCT(5)
      WRITE (6,1018) STPDRQ(6)
      WRITE (6,1006)
      WRITE (6,1010) RQSTCT(2)
      WRITE (6,1011) RQSCH (2)
      WRITE (6,1012) SHORTS(2)
      WRITE (6,1014) PSHORT(2)
      WRITE (6,1016) BUMPED(2)
      WRITE (6,1018) STPDRQ(2)
      WRITE (6,1020) SPBUMP(2)
      WRITE (6,1025) RQSTCT(1)
      WRITE (6,1026) RQSCH(1)
      WRITE (6,1027) BUMPED(1)
      WRITE (6,1028) RQSCH(2)
      STUDRQ(1)= STUDRQ(1)+ STUDRQ(3)+ STUDRQ(4)
      STPDRQ(1)= STPDRQ(1)+ STPDRQ(3)+ STPDRQ(4)
      SPBUMP(1)= SPBUMP(1)+ SPBUMP(1)+ SPBUMP(4)
      SRBUMP(1)= SRBUMP(1)+ SRBUMP(1)+ SRBUMP(4)
      SRSCH(1)= STUDRQ(1)- SRBUMP(1)
      SPSCH(1)= STPDRQ(1)- SPBUMP(1)
      PSPR= SPSCH(1)/ STPDRQ(1)
      PSPS= SPBUMP(1)/ STPDRQ(1)
      WRITE(6,1030)
      WRITE(6,1032) STPDRQ(1)
      WRITE(6,1034) SPSCH(1)
      WRITE(6,1036) SPBUMP(1)
      WRITE(6,1038) SPSCH(2)
      WRITE(6,1040) STUDRQ(1)
      WRITE(6,1042) SRSCH(1)
      WRITE(6,1044) SRBUMP(1)
      WRITE(6,1046) SRSCH(2)
      WRITE(6,1050) PSPR
      WRITE(6,1052) PSPS
      CALL EXIT
      END
/*
//GO.FT16F001 DD DSNAME=+REQUEST,DISP=(OLD,DELETE)
//GO.FT17F001 DD DSNAME=+POINTERS,DISP=(OLD,DELETE)
//GO.FT29F001  DD  DUMMY
//GO.FT40F001 DD DUMMY
//TECHSORT EXEC PROC=DISCSORT
//SORT.SORTIN DD DSN=+TECHINF,DISP=(OLD,DELETE)
//SORT.SORTOUT DD DSN=+TECHOUT,DISP=(,PASS),UNIT=SYSDA,                  X
// SPACE=(TRK,(20,5),RLSE),DCB=(RECFM=VB,LRECL=24,BLKSIZE=244)
//SORT.SYSIN DD *
 SORT FIELDS=(5,2,CH,A)
/*
//TECRYT EXEC PGM=TECRYT
//TECRYT.FT06F001 DD DSN=OUTPUT,UNIT=TAPE,DISP=(MOD,PASS),               X
// DCB=(RECFM=FBA,BUFNO=2,LRECL=133,BLKSIZE=1330)
//TECRYT.FT08F001 DD *        TEACHER INFORMATION FOLLOWS HERE
WED APR 12,1967
     TEST RUN TWO
         ON BYU
/*
//TECRYT.FT14F001 DD DSN=+TEACHRNM,DISP=(OLD,DELETE)
//TECRYT.FT21F001 DD DSN=+TECHOUT,DISP=(OLD,DELETE)
/*
//
```

Job Step 15

STUSTAT

```
C THIS PROGRAM PRINTS OUT THE STUDENT STATISTICS
      IMPLICIT INTEGER*2(A-Z),INTEGER*4($)
      REAL*8 NAME(3),SAM,NAMESS
      DIMENSION INPUT(19),TOTAL(19)
 1000 FORMAT(' ',3A8,2X,7I10)
 2000 FORMAT('1NAME',22X,'REQUESTED  ','SCHEDULED ','FREE      ',
     1 'MAY PDS   ','FULL ATEND','SHORT',5X,'DENIED    ')
      COMMON /BLOK10/NAMESS(141),SAM(3),AV11,AV12,AV13,AV14,AV15,AV16
      DEFINE FILE 13(54,282,U,AV13)
      WRITE(6,2000)
      AV13=0
 1    READ(40) INPUT
 5    TOTAL(1)=INPUT(1)
      DO 10 I=13,19
 10   TOTAL(I)=INPUT(I)
 15   READ(40,END=90) INPUT
      IF(INPUT(I).NE.TOTAL(I)) GO TO 80
      DO 20 I=13,19
 20   TOTAL(I)=TOTAL(I)+INPUT(I)
      GO TO 15
 80   CALL SNAM(TOTAL(1))
      DO 85 I=1,3
 85   NAME(I)=SAM(I)
      TOTAL(16)=28-TOTAL(14)
      WRITE(41) TOTAL
      WRITE(6,100) TOTAL
      GO TO 5
 90   END FILE 41
      CALL EXIT
      END
```

Appendix C

Daily Demand Computer Scheduling Program IBM 360/50
De-Bugging Routines

```
C THIS PROGRAM PRINTS OUT THE REQUEST FILE AFTER PREPARE IS DONE
      IMPLICIT INTEGER*2(A-Z)
      DEFINE FILE 16(600,162,L,AV16)
      DEFINE FILE 17(8,625,U,AV17)
      DIMENSION POINT(2500),A(81)
      READ(17'1) POINT
      DO 10 K=1,2500
      IF(POINT(K).EQ.0) GO TO 20
   10 CONTINUE
      K = 200
   20 WRITE(6,100) (POINT(I),I=1,K)
      READ (17'AV17) POINT
      WRITE(6,100) (POINT(I),I=1,K)
      READ(17'AV17) (POINT(I), I=1,500)
      WRITE (6,100) (POINT(I),I=1,200)
      READ(17'1) POINT
      M = 0
   30 M = M + 1
      IF(POINT(M).EQ.0) CALL EXIT
      READ(16'POINT(M)) A
      WRITE(6,100) A
      GO TO 30
  100 FORMAT (' ',25I5)
      END
```

```
C     THIS PROGRAM PRINTS OUT THE REQUEST FILE AFTER SCHEDULEING
      IMPLICIT INTEGER *2 (A-Z)(ALSO FILES AFFECTED BY SCHEDULING)
      REAL*8 NAMES(500),SNAMES(3,2500)
      DIMENSION RQDATA(81),FILEA(2500),POINT(2500),FILEB(2419),CRSE(2500
     *  ),STDLST(2500),HOLD(28,64)
      EQUIVALENCE(NAMES(1),SNAMES(1,1)),(CRSE(1),SNAMES(1,1000))
      EQUIVALENCE(RQDATA(1),FILEA(1)),(FILEA(82),FILEB(1))
      EQUIVALENCE(POINT(1),STDLST(1))
      DEFINE FILE 11(8,250,U,AV11)
      DEFINE FILE 13(54,262,U,AV13)
      DEFINE FILE 16(600,162,L,AV16)
      DEFINE FILE 17(8,625,U,AV17)
      DEFINE FILE 25(40,896,U,AV25)
 1000 FORMAT('0',5X,'REQUEST NUMBER',I5,5X,'COURSE NUMBER',I5,5X,A8)
 2000 FORMAT(' ',20I6)
 3000 FORMAT(' ',I5,5X,3A8,5X,'STUDENT NUMBER',I5)
 4000 FORMAT(' ',28I4)
      AV25= 1
      READ(11'1) NAMES
      READ(17'1) POINT
      READ(17'3) CRSE
      DO 100 I=1,2500
      A= POINT(I)
      IF(A.EQ.0) GO TO 200
      AV16= A
      READ(16'AV16) RQDATA
      NUM= RQDATA(29)- 50
      IF(RQDATA(28).GT.1) NUM= NUM+ 28
      IF(NUM.GT.0) READ(16'AV16) (FILEB(J),J=1,NUM)
      NUM= NUM+81
      B=CRSE(I)
      WRITE(6,1000) I,B,NAMES(B)
      WRITE(6,2000)(FILEA(J),J=1,NUM)
  100 CONTINUE
  200 READ(13'1) SNAMES
      READ(17'7) STDLST
      READ(25'AV25) HOLD
      DO 300 I=1, 2500
      A= STDLST(I)
      IF(A.EQ.0) GO TO 300
      IF(A.LE.(AV25-1)*64) GO TO 250
      READ(25'AV25)HOLD
  250 WRITE(6,3000)A,(SNAMES(J,I),J=1,3),I
      K= A-(((A-1)/64)*64)
      WRITE(6,4000)(HOLD(J,K),J=1,28)
  300 CONTINUE
      CALL EXIT
      END
```

Job Control Language

Execution Sequence

```
//FLXLIB JOB (FLX,464,002),'QUIST',MSGLEVEL=1,CLASS=D,MSGCLASS=C
//FLXLIB EXEC PGM=IEHMOVE
//FLXLIB.DD1 DD UNIT=TAPE,VOL=SER=9Y2403,DISP=(OLD,KEEP)  UNLOADED PDS
//FLXLIB.DDI DD VOL=SER=DRES02,DISP=(OLD,PASS),UNIT=2314
//FLXLIB.SYSUT1  DD UNIT=SYSDA,SPACE=(TRK,(100,20))
//FLXLIB.SYSPRINT DD SYSOUT=A
//FLXLIB.SYSIN DD *
  COPY PDS=FLXSKED,FROM=2400-3=9Y2403,TO=2314=DRES02
/*
//
//FLXALL JOB (FLX,464,002),'QUIST',MSGLEVEL=1,CLASS=D,MSGCLASS=C
//JOBLIB DD DSN=FLXSKED,VOL=SER=DRES02,UNIT=SYSDA,DISP=SHR
//COPYIN EXEC PGM=COPYIN
//COPYIN.FT01F001 DD DSN=+COURSENM,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),   X
// SPACE=(TRK,3),DISP=(,PASS)
//COPYIN.FT02F001 DD DSN=+ROOMNAME,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),   X
// SPACE=(TRK,2),DISP=(,PASS)
//COPYIN.FT03F001 DD DSN=+STUDNTNM,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),   X
// SPACE=(TRK,18),DISP=(,PASS)
//COPYIN.FT04F001 DD DSN=+TEACHRNM,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),   X
// SPACE=(TRK,4),DISP=(,PASS)
//COPYIN.FT05F001 DD DUMMY
//COPYIN.FT06F001 DD DUMMY
//COPYIN.FT07F001 DD DSN=+MASTERLK,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),   X
// SPACE=(TRK,1),DISP=(,PASS)
//COPYIN.FT09F001 DD DSN=MASTER,UNIT=TAPE,DISP=(OLD,KEEP,KEEP),         X
// VOL=SER=9Y1751,LABEL=(,,,IN)          BYU MASTER
//COPYIN.FT10F001 DD DSN=+KORSINFO,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),   X
// SPACE=(TRK,65),DISP=(,PASS)
//CONDENSE EXEC PGM=CONDENSE
//CONDENSE.FT03F001 DD DSN=+STUDNTNM,DISP=(OLD,PASS)
//CONDENSE.FT05F001 DD DUMMY
//CONDENSE.FT06F001 DD DUMMY
//CONDENSE.FT07F001 DD DUMMY
//CONDENSE.FT17F001 DD DSN=+POINTERS,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),X
// SPACE=(TRK,8),DISP=(,PASS)
//CONDENSE.FT25F001 DD DSN=+STUINFO,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2), X
// SPACE=(CYL,8),DISP=(,PASS)
//PREPARE EXEC PGM=PREPARE
//PREPARE.FT05F001 DD DUMMY
//PREPARE.FT06F001 DD DSN=FLX.OUTPUT,UNIT=TAPE,DISP=(,PASS),            X
// DCB=RECFM=FB
//PREPARE.FT07F001 DD DSN=+MASTERLK,DISP=(OLD,PASS)
//PREPARE.FT08F001 DD DSN=REQUEST,UNIT=TAPE,DISP=(OLD,KEEP,KEEP),       X
// VOL=SER=9Y0905,LABEL=(,,,IN)   BYU REQUEST   WED APR 5,1967
//PREPARE.FT10F001 DD DSN=+KORSINFO,DISP=(OLD,PASS)
//PREPARE.FT16F001 DD DSN=+REQUEST,UNIT=SYSDA,DCB=(DSORG=DA,BUFNO=2),   X
// SPACE=(162,600),DISP=(,PASS)
//PREPARE.FT17F001 DD DSN=+POINTERS,DISP=(OLD,PASS)
//PREPARE.FT25F001 DD DSN=+STUINFO,DISP=(OLD,PASS)
```

```
//TESTPREP EXEC PGM=TESTPREP
//TESTPREP.FT05F001 DD DUMMY
//TESTPREP.FT06F001 DD DSN=FLX.OUTPUT,UNIT=TAPE,DISP=(MOD,PASS),         X
// DCB=RECFM=FB
//TESTPREP.FT07F001 DD DUMMY
//TESTPREP.FT16F001 DD DSN=+REQUEST,DISP=(OLD,PASS)
//TESTPREP.FT17F001 DD DSN=+POINTERS,DISP=(OLD,PASS)
//TESTPREP.FT25F001 DD DSN=+STUINFO,DISP=(OLD,PASS)
//SCHEDULE EXEC PGM=SCHEDULE                                 DE-BUG
//SCHEDULE.FT06F001 DD DSN=FLX.OUTPUT,UNIT=TAPE,DISP=(MOD,PASS),         X
// DCB=(RECFM=FB,BUFNO=2)
//SCHEDULE.FT16F001 DD DSN=+REQUEST,DISP=(OLD,PASS)
//SCHEDULE.FT17F001 DD DSN=+POINTERS,DISP=(OLD,PASS)
//SCHEDULE.FT20F001 DD DSN=+MAYDATA,UNIT=SYSDA,DISP=(,PASS),             X
// DCB=(BUFNO=2,RECFM=VB,LRECL=24,BLKSIZE=244),SPACE=(TRK,(50,5),RLSE)
//SCHEDULE.FT24F001 DD DSN=+RUMARY,UNIT=SYSDA,DISP=(,PASS),              X
// DCB=(RECFM=V,LRECL=2004),SPACE=(TRK,5)
//SCHEDULE.FT25F001 DD DSN=+STUINFO,DISP=(OLD,PASS)
//SCHEDULE.SYSUDUMP DD SYSOUT=D
//MAYSORT EXEC PROC=DISCSORT
//SORT.SORTIN DD DSN=+MAYDATA,DISP=(OLD,DELETE)
//SORT.SORTOUT DD DSN=+MAYOUTA,UNIT=SYSDA,DISP=(,PASS),                  X
// DCB=(RECFM=VB,LRECL=24,BLKSIZE=244),SPACE=(CYL,(2,3),RLSE)
//SORT.SYSIN DD *
 SORT FIELDS=(5,4,CH,A)
/*
//WRYOUT EXEC PGM=WRYOUT
//WRYOUT.FT03F001 DD SYSOUT=E,SPACE=(CYL,(20,10),RLSE),                  X
// DCB=(BUFNO=2,RECFM=FBA,LRECL=133,BLKSIZE=798)
//WRYOUT.FT06F001 DD DSN=OUTPUT,UNIT=TAPE,DISP=(,PASS),                  X
// DCB=(RECFM=FBA,BUFNO=2,LRECL=133,BLKSIZE=1330)
//WRYOUT.FT08F001 DD *          MESSAGE CARDS FOLLOW HERE
WED APR 12,1967
MESSAGE CARD 1
MESSAGE CARD 2
MESSAGE CARD 3
MESSAGE CARD 4
MESSAGE CARD 5
MESSAGE CARD 6
MESSAGE CARD 7
MESSAGE CARD 8
MESSAGE CARD 9
MESSAGE CARD 10
MESSAGE CARD 11
MESSAGE CARD 12
/*
//WRYOUT.FT11F001 DD DSN=+COURSENM,DISP=(OLD,DELETE)
//WRYOUT.FT12F001 DD DSN=+ROOMNAME,DISP=(OLD,DELETE)
//WRYOUT.FT13F001 DD DSN=+STUDNTNM,DISP=(OLD,DELETE)
//WRYOUT.FT14F001 DD DSN=+TEACHRNM,DISP=(OLD,PASS)
```

```
//WRYOUT.FT15F001 DD DSN=+MASTERLK,DISP=(OLD,DELETE)
//WRYOUT.FT16F001 DD DSN=+REQUEST,DISP=(OLD,PASS)
//WRYOUT.FT17F001 DD DSN=+POINTERS,DISP=(OLD,PASS)
//WRYOUT.FT20F001 DD DSN=+MAYOUTA,DISP=(OLD,DELETE)
//WRYOUT.FT21F001 DD DSN=+TECHINF,DISP=(,PASS),UNIT=SYSSQ,                X
// SPACE=(TRK,(20,5),RLSE),DCB=(RECFM=VB,LRECL=24,BLKSIZE=244)
//WRYOUT.FT23F001 DD DSN=+MAYINF,UNIT=SYSDA,DCB=DSORG=DA,DISP=(,PASS), X
// SPACE=(TRK,(2,2),RLSE)
//WRYOUT.FT24F001 DD DSN=+RUMARY,DISP=(OLD,DELETE)
//WRYOUT.FT25F001 DD DSN=+STUINFO,DISP=(OLD,PASS)
//WRYOUT.SYSUDUMP DD SYSOUT=H,SPACE=(CYL,(2,3),RLSE)
//STATS  EXEC PROC=FORTGCLG
//FORT.SYSIN DD *
```